



Thumb Index

Read Me First	
Using The Manual.....	
Setting Up The Hardware	
A Quick Tour	
File Requester	
Sequencer Page.....	
Bar Editor	
Event Editor	
Filters Page.....	
Keymap Editor	
Amiga Samples Page.....	
Librarian Page.....	
Protocol Editor.....	
Tutorials	
Advanced Users	
Glossary.....	
Appendices	
Index	

Music-X™ is a registered trademark of MicroIllusions.

Music-X and the Music-X User's Manual are copyright © 1989 MicroIllusions.

Aldus PageMaker® is a registered trademark of Aldus Corporation.

Amiga™, Intuition™, Workbench™, and Kickstart™ are trademarks of Commodore-Amiga, Inc.

Aztec C™ is a trademark of Manx Software Systems.

Deluxe Paint II™ is a trademark of Electronic Arts.

Flow™ is a trademark of New Horizons, Inc.

KORG™ is a trademark of Keio Electronic Laboratory Corporation.

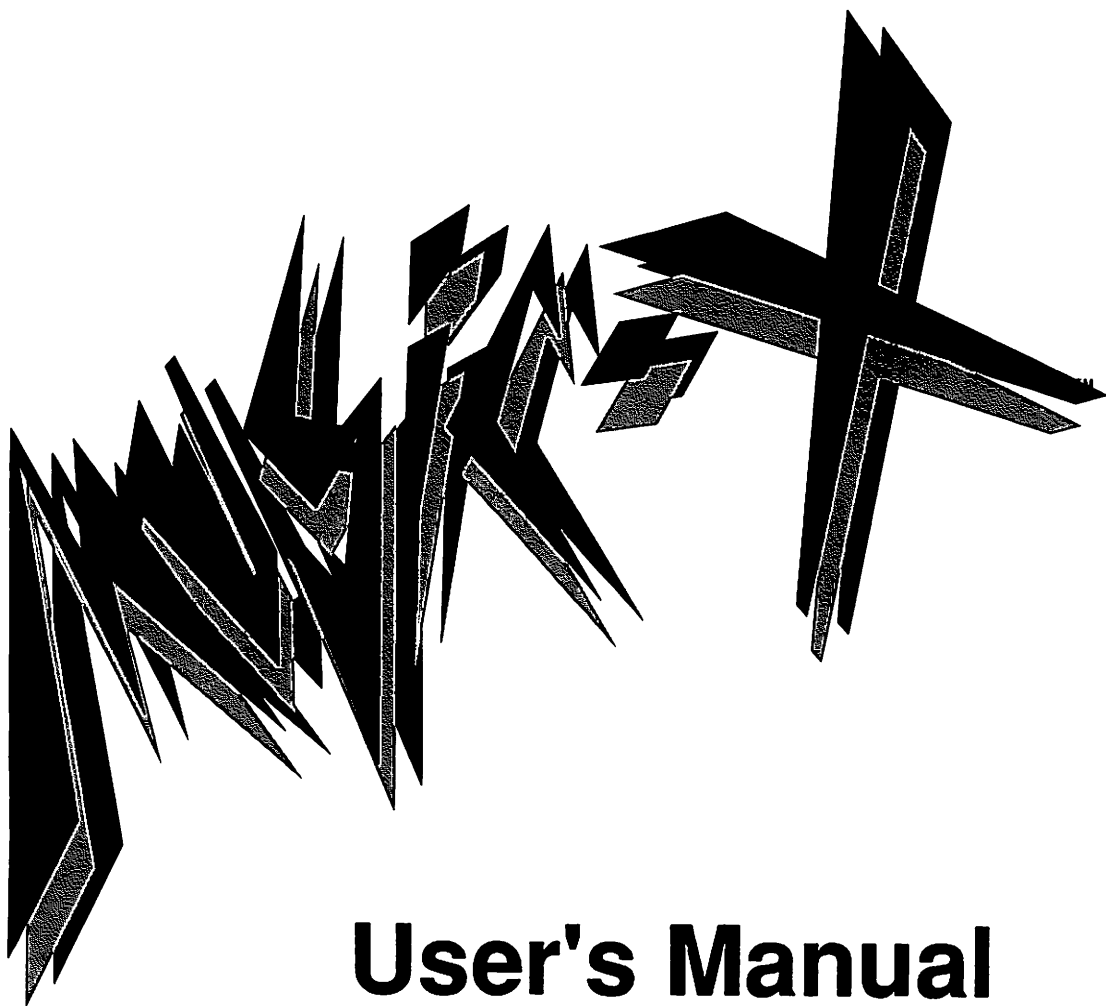
Macintosh™ is a trademark of Apple Computer, Inc.

QEDit™ is a trademark of Prospect Software, Inc.

Roland D-50™ is a trademark of Roland.

Sonix™ is a trademark of Aegis Development, Inc.

Yamaha DX21™ and DX7™ are trademarks of Yamaha Music Corporation.



User's Manual

by Matt Nathan



17408 Chatsworth St., Granada Hills, CA 91344



Table of Contents

IntroductionX

Chapter 1 - Read Me First3

 Recommended equipment3

 Making a working copy on a single-drive system.....4

 Making a working copy on a dual-drive system5

 Rename the disk6

 Reboot7

 Don't copy for friends7

 How to prepare a data disk8

 Save often and make backups8

 Sharing the Amiga with other programs.....9

Chapter 2 - Using the Manual11

 Footnotes12

 Keyboard Equivalents.....13

 Thumb Index13

Chapter 3 - Setting Up the Hardware15

 Minimum Amiga set up.....15

 Beginning MIDI set up.....16

 A daisy chain set up18

 Home studio set up20

 Full blown studio set up21

Chapter 4 - A Quick Tour	23
Boot Music X.....	23
Load and play some music.....	24
A look at the Editors	25
A look at the other pages	26
Record your own sequence	28
Saving a Performance	29
 Chapter 5 - File Requester	 31
What are files?	31
What are directories?	32
Requester description	33
How to use the file requester	38
Standard filenames	39
Save often and make backups	42
 Chapter 6 - Sequencer Page	 45
Overview	45
Title Bar	46
Mode Menu	47
File Menu	50
Options Menu	58
Sync Menu	68
Modules Menu	90
Transport Window	91
Record Sequence Window	98
Clock Window	107
Tempo Window	109
Time Signature Window	111
Sequence List Window	112
Track Window	117

Store	118
Edit.....	119
Delete	121
Preview	121

Chapter 7 - Bar Editor..... 126

Editing Concepts	126
Bar Editor Overview	128
Title Bar	129
File Menu	129
Edit menu	134
Options Menu	135
Display Menu	144
Modules Menu	146
Quantizer Module	146
Velocity Scaling Module	152
Aftertouch Scaling Module.....	157
Graphic Score Window.....	161
Event Types.....	166
Tool List	196
SetSig	196
Time Signature Display	197
Add.....	198
Insert.....	200
Move	201
Remove	202
Lock	202
Select	204
Mark	205
Unmark	207
Zoom+ and Zoom-	207
Snap	208

Bar Editor (continued)

Quiet	210
Scroll	211
Grid	212
Params	216
Repeat	220
Seq:	221
T=	222
D=	223
Transport Controls	223
Recording Modes	225
Channel Square	232
STEP and BACK	233
Virtual Sliders	234
Grid: and Dur:	235

Chapter 8 - Event Editor238

Overview	238
Title Bar	241
File Menu	241
Edit menu	242
Options Menu	242
Display Menu	246
Modules Menu	247
The Event List	247
Event Types	250
Tool List	263
SetSig	263
Signature Display	264
Dup	264
Remove	264
Select	265

Unmark	266
Undo.....	266
Quiet	266
Scroll	267
Grid	267
Params	268
Repeat	268
Event Type:	269
Channel:	269
STEP and BACK	269
Seq:	270
Transport Controls	270
Virtual Sliders	271

Chapter 9 - Filters Page273

Overview	273
Title Bar	278
Mode Menu	278
File Menu	278
Modules Menu	280
Channel	280
Remap Panel.	281
MIDI Message Panel	285
Keyboard Map	287
Data Echo	288

Chapter 10 - Keymap Editor293

Overview	293
Title Bar	296
File Menu	296
Mode Panel.....	298
Miniature Keyboard.....	302

Keymap Editor (continued)

Action List	308
Parameter Window	314
MIDI Recording	314

Chapter 11 - Amiga Samples Page.....319

Overview	319
Title Bar	321
Mode Menu	321
File Menu	321
Options Menu	324
Sample List.....	325
Edit Panel	327
Creating Envelopes	332

Chapter 12 - Librarian Page.....335

Overview	335
Title Bar	339
Mode Menu	339
File Menu	340
Options Menu	342
Modules Menu	345
Mode Panel.....	346
Channel:	357
Page Displays	357
Generic Patch Editor	359

Chapter 13 - Protocol Editor.....362

Overview	362
The Face of the Protocol Editor	365
Title Bar	366

File Menu	366
Name Panel	367
RECIEVE Panel	369
SEND Panel	370
SUB-STRINGS Panel	371
Protocol Language	372
 Chapter 14 - Tutorials	 389
Amiga Basics	389
MIDI Overview	395
Setting Up a Metronome Track	396
 Chapter 15 - Advanced Users.....	 399
Editing Tricks	399
Song Assembly	403
Workbench Tool Types	410
Installing Modules	416
File Conversion	419
 Afterword	 420
 Glossary	 422
 Appendices	 438
Keyboard Equivalents.....	438
System Messages	442
Protocol Language Summary	452
Music-X File Format	456
 Index.....	 465
 Bibliography	 479

Introduction

Congratulations on your intelligent choice in professional music software! Music-X is a comprehensive package designed for the professional composer or musician, but you don't have to be a professional to use it. You'll find many innovative features that are not only powerful but intuitive in nature. Music-X was written on the Amiga for the Amiga and takes full advantage of high speed color graphics and multitasking to create a fast and friendly work environment. We hope you will be inspired to experiment and create with this instrument of the future.

The people behind Music-X

Music-X's creator, David Joiner, who can be reached on BIX as "Talin", is a modern renaissance-man type of guy. Painter, composer, award winning costume maker, and programmer (most noted for "Discovery", and his wonderful "Faery Tale Adventure" game). Being a free thinker, Dave is known to say things like, "The only mind altering substance I use is breakfast."

The manual's author, Matt Nathan, balances between producer, computer artist, eclectic philosopher, and LA salsa musician. This is his first effort as a writer. He likes animation, natural foods, weird time signatures, and microtuning. Some of his music can be found on the Music-X Examples disk.

The cover art was done by Allison Hershey, whose unique techno-primitive painting style is gaining kudos at west coast Science Fiction conventions.

Resident video wizard Mike Berro supplied the SMPTE interfacing routines for Music-X.

The Workbench interface routines were supplied by Joe Pearce, a studying astrophysicist, and aficionado of role-playing games. His current project is "Land of Legends".

The file conversion programs on the Music-X Utilities disk were written by Ken Jordan, bit cruncher extraordinaire, and Trina Black, pixel pushing pixie.

Manual illustrations by Matt Nathan and David Joiner.

Manual layout by Tina Gagliano.

Index compiled by Curtis Norris.

Editing by Greg Hemsath.

Special thanks go to all the people who gave us their feedback and suggestions, especially:

Brian Flanagan
Charles Winston Stokes
Chris Arley Seeger
Dan Smith
Dave Quinlan
Glen Deskin
Ira Rubnitz
James Redd
James Saad
James Seeley
Kelly Holthaus
Steve Gillmore
Steve Nolan
Sun Data
Todd Grace
Transcom Digital
Vinnie Tieto

The Future

Music-X will remain a viable package and continue to grow as music technology grows. Two things assure this. First, the commitment on the part of MicroIllusions to ongoing updates and expansions as the scope of music sequencing changes. Second, open-ended modular compatibility. This means Music-X will support third party modules (mini-programs) that link to and extend Music-X. You may even want to write and add on modules of your own!

About our tools

The Music-X User's Manual was originally typed in outline format using the FlowTM outline processor on an Amiga 1000. The illustrations were created using DeluxePaint IITM. Screen dumps were captured using Steve Tibbet's ScreenX utility, and then retouched. The final typesetting was done in Aldus PageMakerTM on a Macintosh II, and printed on a LaserWrite II NTXTM.

The Music-X program was written in a combination of C and assembly language, using the Aztec CTM development system. All source files were created in Matt Toschlog's QEDitTM text editor. All screens and windows were originally visualized using Deluxe Paint IITM.



Chapter 1 - Read Me First

Recommended equipment.	3
Making a working copy on a single-drive system. . . .	4
Making a working copy on a dual-drive system. . . .	5
Rename the disk	6
Reboot	7
Don't copy for friends	7
How to prepare a data disk	8
Save often and make backups	8
Sharing the Amiga with other programs	9

Read Me First

Even if you read nothing else, read this chapter. Before you start to use Music-X, there are some important things that must be done. This chapter walks you through them. It tells you how to protect your original disks and make working copies. It also explains a few things you'll need to know about the Amiga and Music-X before we embark.

Recommended equipment

You need these to run Music-X using the internal voices:

- 1 Amiga with a minimum 512K RAM (1 megabyte RAM recommended)
- 1 monitor with audio

To use MIDI you need these:

- 1 MIDI interface
- 1 MIDI keyboard-synthesizer or sound module

Recommended to realize the full potential of Music-X:

- As much RAM as you can get
- A second disk drive
- A hard drive
- Racks of MIDI devices

Your Music-X package should come with:

- 1 Music-X Program disk
- 1 Music-X Examples disk
- 1 Music-X Utilities disk
- 1 User's manual
- 1 MicroIllusions Warranty Registration card

If the three disks are not already write-protected, then protect them by sliding the black protect-tab open on each of them.

Making a working copy on a single-drive system

First of all, you'll need some new blank disks to copy onto. Then follow these steps:

- ☐ Once you've loaded Kickstart (A500 and A2000 owners needn't worry about that since Kickstart is in ROM) and your computer is asking for a Workbench disk, insert the Music-X program disk.
- ☐ The Workbench screen will appear. Along the top is the title bar stating which version of the Amiga's Workbench is being used and how much free memory is available in your system. Along the right edge you'll see some disk "icons" (an icon is a picture which you can "click" with the mouse).
- ☐ Up in the left corner you'll see the mouse pointer. You move this by moving the mouse.
- ☐ Move the mouse pointer over to the disk icon named **Music-X**.
- ☐ "Click" on the icon once (while the pointer is positioned over the icon, press the LEFT mouse button and release it).
- ☐ The icon will become highlighted.
- ☐ While the Music-X disk icon is highlighted, press and hold the RIGHT mouse button.
- ☐ The title bar will change to a list of menu names.
- ☐ While still holding the right mouse button, move the mouse pointer to the word **Workbench** in the menu bar. The Workbench menu will pull down.
- ☐ Continuing to hold the RIGHT mouse button, move the mouse pointer down through the list.
- ☐ Stop at **Duplicate** and release the RIGHT mouse button. You've just given the Amiga the command to duplicate a disk.
- ☐ A requester will pop up titled **Disk Copy**. Follow the directions. You'll be asked to put the blank disk in a drive. You'll be asked to exchange the original and copy disks a few times. IMPORTANT! Never remove a disk from a disk drive while the red drive light is on. Doing so can destroy your disk and damage your disk drive.

Making a working copy on a dual-drive system

First of all, you'll need some new blank disks to copy onto. Then follow these steps:

- ❑ Once you've loaded Kickstart (A500 and A2000 owners needn't worry about that since Kickstart is in ROM) and your computer is asking for a Workbench disk, insert the Music-X Program disk into drive DF0: (the internal drive).
- ❑ The Workbench screen will appear. Along the top is the title bar stating which version of the Amiga's Workbench is being used and how much free memory is available in your system. Along the right edge you'll see some disk "icons", one of which says **Music-X** (an icon is a picture which you can "click" on with the mouse).
- ❑ Insert a blank disk into the second disk drive in your Amiga system. Normally this would be DF1:. However, if you have an A2000 with one internal drive and one external drive, the second (external) drive would be DF2:.
- ❑ There should now be three disk icons on the screen: one that says **Music-X**, one that says **RAM DISK**, and one that says **DF1:BAD**.
- ❑ In the upper-left corner you'll see the mouse pointer. You move this by moving the mouse. Move the mouse pointer over to the disk icon named **Music-X**, and press and hold down the left mouse button. You'll find that if you move the mouse around while holding down the left mouse button, you can "drag" an icon around the screen.
- ❑ Drag the **Music-X** icon over until it is directly on top of the **DF1:BAD** icon, and release the left mouse button. Instructions will appear on the Workbench screen telling you what to do from here.

Rename the disk

You've now duplicated your program disk. There's one more step before the working copy can be used; it must be renamed from "copy of Music-X" to "Music-X". (This is because Music-X expects to find itself on a disk called "Music-X".)

- ☐ Put the working copy disk back into a drive, if it's not already there. Click once with the left mouse button on the icon named **copy of Music-X**. It should become highlighted. Move the mouse pointer off the icon.
- ☐ Press and hold the RIGHT mouse button.
- ☐ The title bar will change to a list of menu names.
- ☐ While still holding the right mouse button, move the mouse pointer to the word **Workbench** in the menu bar. The Workbench menu will pull down.
- ☐ Continuing to hold the right mouse button, move the mouse pointer down through the list and choose **Rename** by releasing the right mouse button when that word is highlighted.
- ☐ A "text-box" will appear containing the disk's name, i.e., "copy of Music-X". A "cursor" will be highlighting the first character. Press the Amiga's DEL (delete) key repeatedly to remove the unneeded portion of the name "copy of " (be sure to delete the space after the word "of"—the cursor should be on the letter "M"). When the name again reads "Music-X", press the RETURN key and wait until the drive light goes out. The disk is renamed.

Reboot

Duplicate and rename your other disks in the same manner.

When done, to avoid any confusion between the old and new “Music-X” disks, you should now reboot (restart) the computer using the working copy.

- ☐ Remove the original disks from the drives and put them back in a safe place. (If you ever have a problem with a working copy, you can follow the previous steps again.)
- ☐ Simultaneously press, and then release the three keys: CTRL, LeftAmiga, and RightAmiga. When the Amiga asks for a Workbench disk, insert the working copy. Consider this your Music-X disk from now on. The Amiga keys are the ones to either side of the space bar with the large Amiga “A” or the Commodore symbol on them (depending upon which model Amiga you have).

Don't copy for friends

Your copies are for you only. The true professional can't afford to lose time if his or her program disk fails in the studio. For this reason Music-X was left un-copyprotected. Please accept this gift of vulnerability with honor. Be fair and don't make copies for your friends or others.

How to prepare a data disk

You'll need some new empty disks to put your new performance data onto as you create.

You prepare a new disk so that the Amiga can read and write data from it by "initializing" it. To do that, put a blank disk in a drive. Then, from the Workbench, click on the disk's icon once to highlight it. Then choose **Initialize** from the **Disk** menu, and follow the directions. The disk will be automatically named "Empty". You may wish to rename it to something more descriptive, like "Songs" or "Music Data".

Save often and make backups

There are two habits you can keep, that may prevent the accidental loss of hours of work and inspiration.

First, you should periodically save to disk the music you create during a work session. Don't wait until you're done. The more often you save, the better. Save whenever a moderate improvement has been made to the data. That way, if there's a power outage, or you change your mind after deleting something, you'll be able to reload the most recently saved version from disk and continue working.

Second, when you're done with the session, make a copy of the disk that you've been saving to. This provides a backup in case something happens to the original.

These ideas can't be stressed enough. When using computers you are working with data in a wonderfully plastic way. However, magnetic mediums are fragile and accidents can happen. So start the habits of saving often while you work, and making backups after each session.

Sharing the Amiga with other programs

Music-X reserves most of the Amiga's available memory for its own potential use when it boots up. If you plan to use other programs at the same time, then you should load them first before you load Music-X.

If too many programs are loaded, and the Amiga runs out of memory, then Music-X may not be able to load its own Modules. Nothing dangerous will happen (the Amiga's operating system will simply return to Music-X after an unsuccessful load), but things like the Quantizer Module may not be available. If you need more memory, quit Music-X, close the other programs, and reopen Music-X, so it has room to function.

A Modules menu appears in each of Music-X's main pages and in the Editor pages. This is where access is given to externally loaded Modules. The first time you use a Module it is loaded from disk, after that it tends to stay in memory. In systems with less memory, a module may be erased (from RAM, not disk) when a new module is requested. This is called "overlaying". When you call a module you may be asked to put a particular disk in a drive. Remember that since loading from disk takes time, the first appearance of a module will not seem as instantaneous as following appearances.

If you plan to have other programs in memory at the same time as Music-X and you want to reserve room for them, there is an alternative to loading them first. You can preset the amount of memory that Music-X reserves when it boots up, by defining the buffer size in the info files for either Music-X or its performance files. This is explained in the Advanced Users chapter.

[see Advanced Users chapter / Workbench Tool Types]

Chapter 2 - Using the Manual

Using the manual 11

Footnotes12

Keyboard Equivalents13

Thumb Index13

Using The Manual

You should find Music-X to be mostly self-explanatory. All functions are available on the screen or from menus using the mouse, most having an equivalent keyboard command as well. It should be easy to keep an intuitive work flow going without having to worry about obscure control codes or hidden features.

However, when you want more detail, or if you feel stuck, you'll find that this manual has been designed to keep you moving as painlessly as possible. Think of this as a "real time" manual to be accessed while you work. Every effort has been made to make it easy for you to get whatever information you need as quickly as possible. Each of the different working screens in Music-X (called "Pages") has it's own chapters, and there are reference chapters in the back.

Here are some situations when you might want to use the manual, and an approach to follow for each:

- ☐ You're stuck in a function and can't get it to do what you thought it was supposed to do. – The quickest way to find out what to do next is to look in the Table of Contents for the current Music-X Page and function you're in. Then turn to the referenced page for a detailed description of the function you're using and how it relates to other functions.
- ☐ You feel sort of lost about where to start. – A tutorial section has been written to walk you through the steps in basic uses of the program.
- ☐ You know what you want to do but you don't know how to get there from where you are. – The best thing to do is look in the Index for an alphabetical listing of all the possible functions. When you find one that might be applicable, turn to the noted pages for further info.
- ☐ You come upon a word or term you don't understand. – Simply turn to the Glossary for short descriptions of the special terms used in this manual.

- ❑ You want to make sure you didn't miss any of the cool stuff the program does just because you haven't run across it with the mouse yet. — At some point, you should read the manual completely to get the most use and satisfaction out of the software.

Once you've become familiar with the software you can use the Appendices at the back of the manual for quick reference.

Footnotes

Footnotes are given throughout the manual to direct the reader to other sections that can shed light on the current subject. The format for footnotes is [Chapter / Sub Chapter / Paragraph Heading / Sub Paragraph Heading]. This format is used because it's analogous to the actual software. As you'll see later, one can just as easily find a function in the software as find its description in the manual. This format is also used throughout the manual, in the header at the top of each page, to show where one is in the manual. Watching these headings while one thumbs through the manual can quicken the search for a referenced function.

For example: “[see Using the Manual / Footnotes]” would direct you here.

Lets examine this footnote, “[see Bar Editor / Params / Time Signature]” in greater detail.

To find the referenced section of the manual, the footnote first directs you to look in the “Bar Editor” chapter for the “Params” sub chapter. Within the Params subchapter would be the paragraph describing the “Time Signature” display.

To find the same function in the software, go to the Bar Editor Page. Look in the list of tools for **Params** and click it. When the **PLAY PARAMETERS** window appears, look for the Time Signature display.

Keyboard Equivalents

Many functions in Music-X can be initiated from the Amiga keyboard as well as from the screen controls. This is usually accomplished by pressing one key, or a combination of two keys together. This is called the “keyboard equivalent”. Manual descriptions for screen functions that have keyboard equivalents will begin with a notation showing the key, or key combination, that is equivalent to using the screen control. An example of this would be:

(RightAmiga-“Q” from the keyboard)

This is the keyboard equivalent that quits Music-X. What the example is telling you is that the right “Amiga” key should be held while the “Q” key is pressed. The Amiga keys are the ones to either side of the space bar, with the letter “A” or the Commodore symbol on them, depending upon which model Amiga you have.

Thumb Index

The first page in the manual contains the “thumb index” which can be used to find chapters quickly. Along the right edge of the page is a column of markers that extend to the edge of the page. Each page within a listed chapter is also marked with one of these.

To find a chapter quickly, look in the thumb index and note the position of the marker next to the heading of the chapter you want, then thumb through the manual until you find the pages having markers in the same position.

Chapter 3 - Setting Up the Hardware

Setting up the hardware	15
Minimum Amiga set up	15
Beginning MIDI set up	16
A daisy chain set up	18
Home studio set up	20
Full blown studio set up	21

Setting up the hardware

Since MIDI was basically created to allow people to assemble their own modular music systems, there's no single, "normal" equipment set up; every system is expandable. However, this chapter will describe some of the basic types of equipment set ups that can be used.

Minimum Amiga set up

Illustration 3.1 shows the use of Music-X on the most basic Amiga with one drive. In this case, no extra MIDI equipment is required to play songs with Music-X. All sounds are played internally using Amiga samples and heard through the Amiga monitor. Music can be written manually, using the sequence editors, but not recorded live. To do that we need a MIDI keyboard.

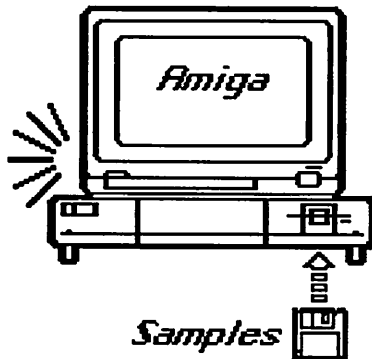


Illustration 3.1 Amiga Solo

Beginning MIDI set up

To use MIDI equipment with the Amiga, we first need a MIDI interface. There are various interfaces available for the Amiga, and some are very inexpensive. Some interfaces give you simple MIDI IN and OUT. Others offer RS-232 pass through to accommodate other devices that normally use the serial port (printer, MODEM, etc.). Still other interfaces are more like “thru boxes”, and have multiple INs and OUTs.

◆ Note: If you use a MIDI interface with RS-232 pass-through feature, make sure that the RS-232 switch is in the MIDI position when Music-X is first booted. This will avoid any potential problems caused by Music-X trying to read MIDI information from a MODEM or other non-MIDI device.

MIDI interfaces plug into the serial port. The A1000 uses a picture of a telephone to identify this port, the other Amiga models say “SERIAL”. Follow the installation directions that come with the particular model interface you buy. Throughout the manual, references to the Amiga’s MIDI IN and MIDI OUT actually refer to the Amiga’s interface’s MIDI IN and MIDI OUT.

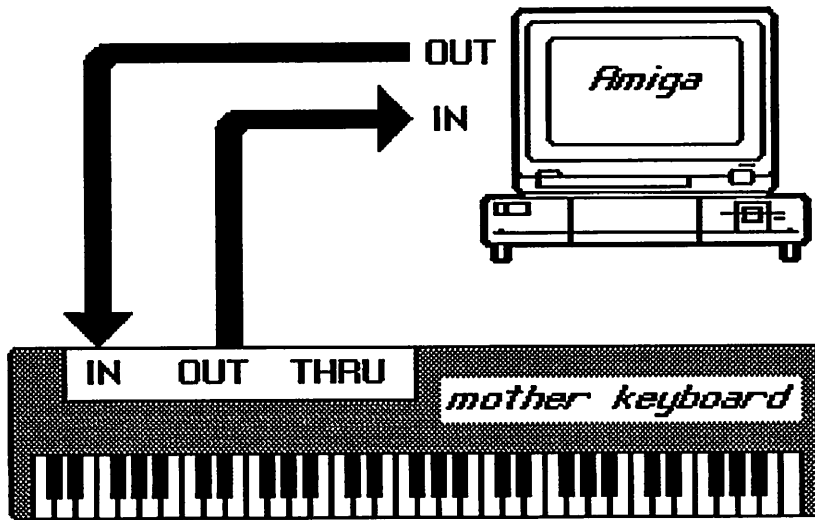


Illustration 3.2 Simple MIDI Setup

Illustration 3.2 shows the introduction of a MIDI keyboard into the system. With this “mother keyboard” connected to Music-X we can record and play back MIDI data. To connect the mother keyboard:

- ☐ Connect a MIDI cable from the keyboard’s MIDI OUT to the Amiga’s MIDI IN. The keyboard sends information to the Amiga through this cable.
- ☐ Connect a MIDI cable from the Amiga’s MIDI OUT to the keyboard’s MIDI IN. The Amiga sends information to the keyboard through this cable.
- ☐ Connect the audio output of the keyboard to an amplifier or headphones.

A daisy chain set up

Illustration 3.3 shows the beginnings of a “daisy chain”. A daisy chain allows the computer to send messages to more than one MIDI instrument at the same time. This is done using the MIDI THRU ports of the MIDI instruments in the chain. The data that an instrument sends out its MIDI THRU port is an exact copy of what the instrument receives at its MIDI IN port. When MIDI instruments are connected in a daisy chain, each instrument passes the data it receives to the next instrument in the chain without changing it; they all get the same data.

The only drawback is that long daisy chains can cause minute timing lags due to the accumulated errors in response time when each instrument re-transmits the data. The longer the chain, the more noticeable the lag. In most cases, with just a few instruments, this effect is imperceptible.

The cables are connected in the daisy chain example the same as in the previous example, with these additions:

- ❑ A MIDI cable is connected from the MIDI THRU of the mother keyboard to the MIDI IN of the next instrument in the chain.
- ❑ To add an instrument onto the end of the chain, connect a MIDI cable from the MIDI THRU of the last instrument in the chain to the MIDI IN of the new instrument.

The data flow in the daisy chain is from the MIDI OUT of the mother keyboard to the MIDI IN of the computer. Then from the MIDI OUT of the computer to the MIDI IN of the mother keyboard. Then from the MIDI THRU of the mother keyboard to the MIDI IN of the next instrument in the chain. Then from the MIDI THRU of that instrument to the MIDI IN of the next instrument, etc.

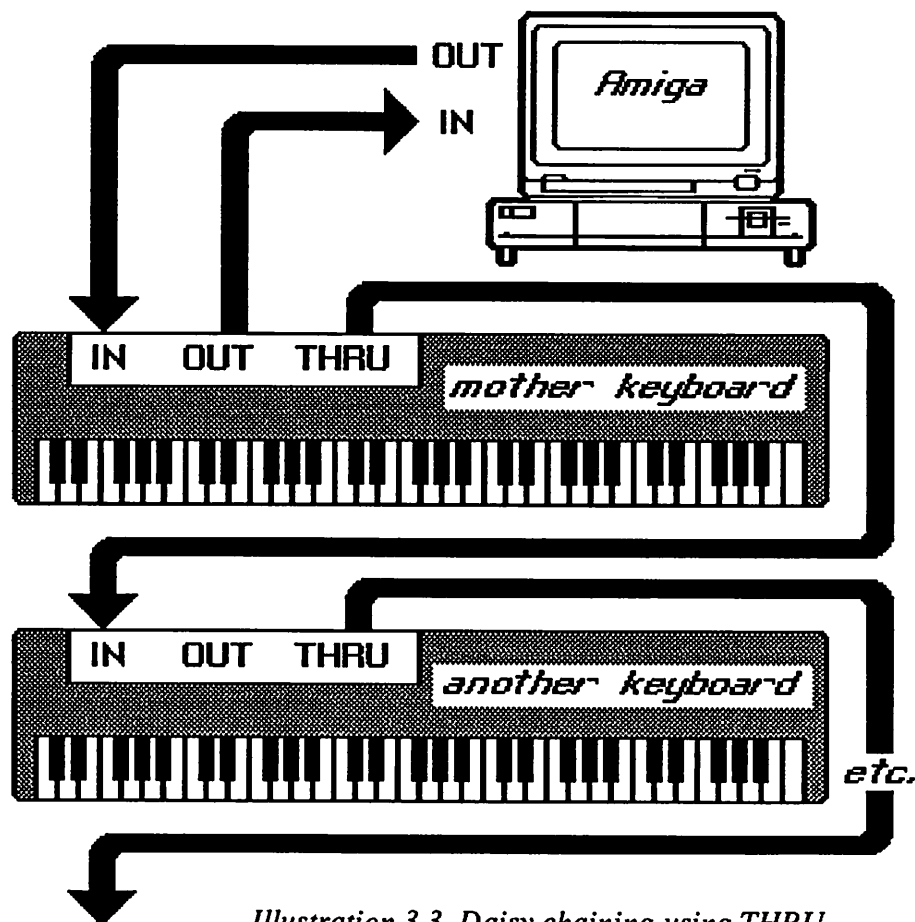


Illustration 3.3 Daisy chaining using THRU

Home studio set up

Illustration 3.4 shows a moderate home studio with a mother keyboard, a drum machine and an external sound module. An expanded MIDI interface has been included to eliminate the daisy chain. The MIDI OUT of the Amiga is duplicated in the multiple MIDI OUT ports of the interface. Each receiving device gets a copy of the same data, with no delay. The switchable MIDI IN ports of the interface allow the mother keyboard or the drum machine to be used as the main input device, without changing cable arrangements. More modules can be added and connected to the unused MIDI OUT ports of the interface as the system grows.

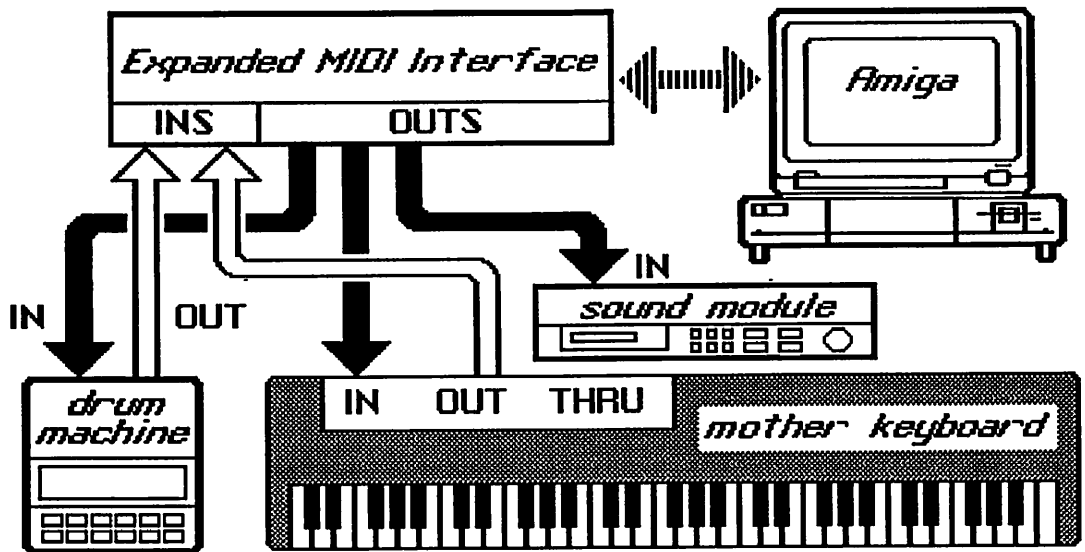


Illustration 3.4 A home MIDI studio

Full blown studio set up

Illustration 3.5 shows an example of a full blown MIDI studio set up. A daisy chain is no longer being used. MIDI data coming from the Amiga is being distributed by the thru-box type MIDI interface to a rack of sound modules and MIDI controlled signal processors. A mother keyboard and a drum pad are both connected to the switchable MIDI IN ports of the interface. The performer can switch between using the mother keyboard and the drum pad to enter music, without having to unplug cables. A SMPTE Reader is being used to synchronize Music-X to Time Code information coming from a 24 track tape recorder. Even this massive set up does not show the limits of a MIDI compatible system. One could still add video or film equipment, MIDI light show equipment, MIDI mergers (to record more than one instrument at once), etc.

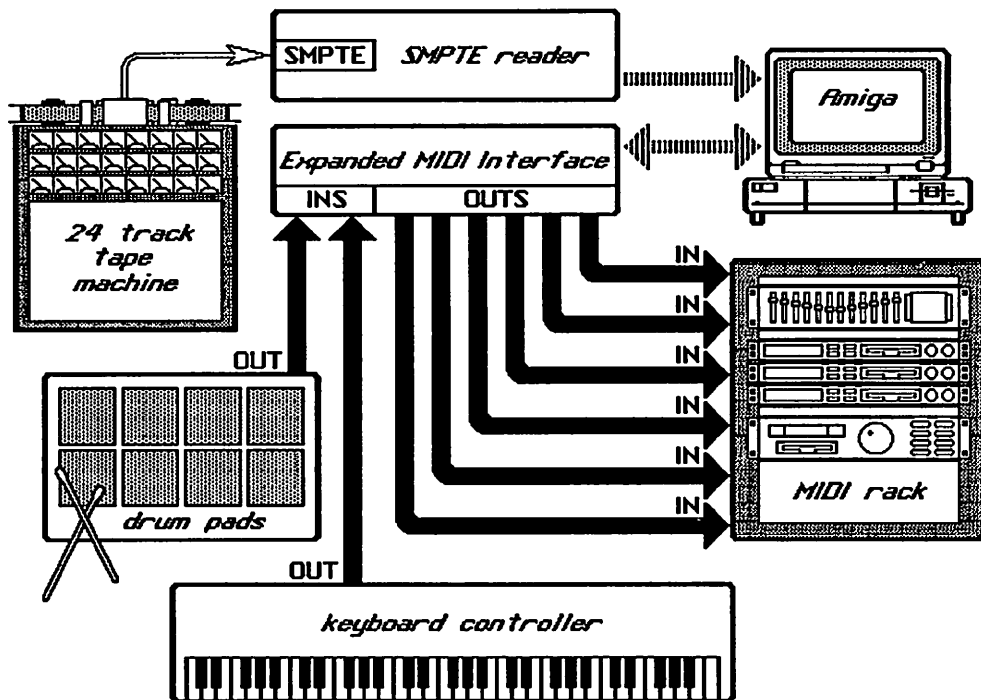


Illustration 3.5 Studio with a whole lot of equipment

Chapter 4 - A Quick Tour

Boot Music-X 23

Load and play some music 24

A look at the Editors 25

A look at the other pages 26

Record your own sequence 28

Saving a Performance 29

A Quick Tour

When you first get a program, do you do this? 1) Pick up the manual thinking “I really should read this first.” 2) Get instantly bored and go ahead and boot the program without knowing anything. 3) Get lost in the program and eventually end up turning off the machine in frustration. 4) Begrudgingly go back to the manual grumbling, “I thought this was supposed to be fun...”

Well, this chapter should move just fast enough to guide you into the program without having to read a lot. One of the most creative times with a new instrument is when you first get it, while the initial burning curiosity is driving you. So, read this chapter, then go ahead, “knock yourself out!” After you’ve had some fun exploring Music-X, please come back and check out the rest of the manual.

If this chapter seems to move too quickly, you might want to read the tutorial on Amiga basics first.

Boot Music-X

First let’s get Music-X up and running. I’m assuming that all the Amiga hardware is properly connected before we start. If you need to know how to do that, read the documentation that came with your Amiga.

- ☐ Turn your Amiga on.
- ☐ Amiga 1000 owners now put the Kickstart disk in the internal drive. When the disk drive light goes out, remove the disk.
- ☐ Put the Music-X program disk in the internal drive and wait for the Workbench to load.

- ☐ Open the Music-X disk by double clicking its icon (it looks like a disk).
- ☐ Load and run Music-X by double clicking its icon (it looks like a miniature cassette player).
- ☐ Music-X will boot up in the Sequencer Page.

Load and play some music

Plenty of sample songs are provided on the Music-X Examples disk. Let's load one and play it.

- ☐ Put the Music-X Examples disk in a drive.
- ☐ Look in the File menu and choose **Load Performance**.
- ☐ When the file requester window appears, click on the drive number that you put the Examples disk in (**DF0:** or **DF1:**).
- ☐ When the list of entries is complete, look for and double click on **Performances (dir)**.
- ☐ When the new list of entries is complete, look for and double click on **Demo1.Perf**.
- ☐ After the file requester goes away, click **BEGIN**.
- ☐ Click **PLAY**.
- ☐ Assuming your Amiga's audio output is hooked up, you'll be hearing the internal voices playing a sequence.
- ☐ Click **STOP** when you're done listening.

A look at the Editors

Now that we've heard it, let's see what the music looks like.

- ☐ Click **EDIT** to move into the Bar Editor Page.
- ☐ Click **PLAY**. The Bar Editor lets you see what you're hearing. Notes appear as colored bars.
- ☐ Click **STOP**.
- ☐ Add a few new notes yourself by clicking the mouse anywhere in the score window. Then click **PLAY** to play them back.
- ☐ While you're here you can see the Event Editor Page as well. Look in the Edit menu and choose **Event**. The Event Editor lists all the events in a sequence. (Notes are one kind of event.)
- ☐ To exit, look in the File menu and choose **Exit**. When the exit prompt appears, click **DISCARD**. This should move you back into the Sequencer Page.

A look at the other pages

Music-X is composed of four main pages and at least four ancillary pages. Illustration 4.1 shows how to move around between the pages in the overall map of Music-X. Let's look at them one by one.

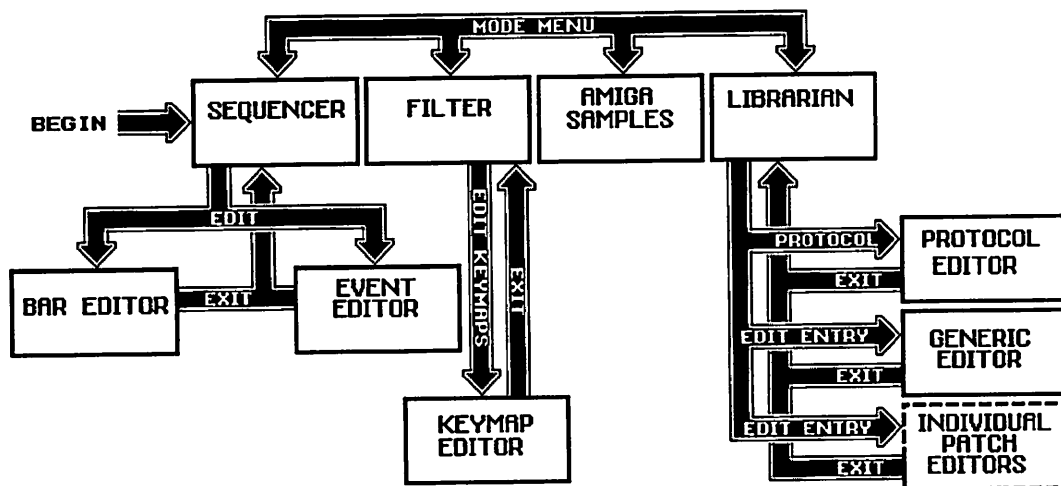


Illustration 4.1 Music-X overall organization

- ❑ Look in the Mode menu and choose **Set Filters**. This will move you into the Filters Page which is used as a MIDI router. Information coming from the mother keyboard can be redirected, selectively thinned out, or ignored altogether using the controls in this page.
- ❑ From the Filters Page you can get to the Keymaps Page. Look in the Modules menu and choose **Edit Keymaps**. (The Keymap editor is a module and is loaded from the program disk.)
- ❑ The Keymap Editor Page creates special MIDI routing maps used by the Filters Page. Notes coming from the mother keyboard can be used to trigger other actions besides the playing of notes; for example: start the sequencer,

stop the sequencer, play a particular sequence, etc. Return to the Filters Page by choosing **Exit** from the File menu.

- ☐ Now choose **Amiga Samples** from the Mode menu.
- ☐ The Amiga Samples Page is used to handle the internal sounds. You should see the sounds listed here that were automatically loaded with **Demo1.Perf**. There is an envelope editor at the bottom of the page.
- ☐ Now choose **Librarian** from the Mode menu. The Librarian Page is used to store programs for synthesizers. A program is an internal setting that a synthesizer uses to create its own sounds. Each program creates a different sound. Most synthesizers have a set number of programs they can remember at any one time. By storing extra programs in Music-X, you can increase the variety of available sounds beyond the capacity of the synthesizer.
- ☐ From the Librarian Page you can get to the Edit Protocol Page. Look in the Edit menu for **Create Protocol**. The Edit Protocol page is used to set up the special protocols that the Librarian Page uses when it sends and receives programs from a synthesizer. A protocol is like a mini-language that only that synthesizer understands. It's here that you teach Music-X how to talk to new synthesizers.
- ☐ Return to the Librarian Page by choosing **Exit** from the File menu.
- ☐ A few Patch Editor pages are accessible from the Librarian Page as well. These are loaded separately, and only when needed. Normally you would use them by first loading a library file from the File menu, and then choosing **Edit Entry** from the Edit menu. Learn more about them in other chapters.
- ☐ Move back to the Sequencer Page by selecting **Sequencer** from the Mode menu. That's how you get around in Music-X. We've covered eight Pages: Sequencer Page, Bar Editor, Event Editor, Filters Page, Keymap Editor, Amiga Samples Page, Librarian Page, and Protocol Editor.

Record your own sequence

Using a MIDI keyboard, let's record and play back a new sequence. The mother keyboard should be hooked up and its "send" and "receive" channels should be set to the same channel. If you need to know how to do this, read the owner's manual for your instrument.

- ☐ Choose **Load Performance** again from the File menu.
- ☐ This time load **SongStart.Perf**.
- ☐ Click **PLAY**, listen to the metronome sequence and adjust the tempo by dragging the tempo slider (found in the left center of the Sequencer Page, and identified by a quarter note symbol followed by an equals sign).
- ☐ When the tempo is right, click **STOP**.
- ☐ Click **BEGIN** to move the clock back to the beginning of the piece.
- ☐ Click **RECORD**. The Record requester will appear.
- ☐ Tap a note on the mother keyboard to start the clock.
- ☐ Wait for the metronome to count in eight beats, then start playing some notes on the mother keyboard, in time with the metronome.
- ☐ When done, click **STOP**. The music you just played is now captured in the "Record Buffer".
- ☐ At the bottom of the Sequencer Page is the "sequence list". The "current sequence" is the one that's highlighted in blue in the list. Select an empty sequence in the sequence list by clicking on it, and click **STORE**. This takes what you just played from the Record Buffer and puts it in a new sequence.

- ☐ Click **BEGIN** and **PLAY** to hear what you've just recorded.
- ☐ If you don't like it, click **DELETE** to erase the current sequence (the one highlighted in the list).

Saving a Performance

If you record something you like, you'll need to copy it from the Amiga's temporary memory into a more permanent form as a file on a disk.

- ☐ Put a writable disk in a drive. (If you have more than one drive, any drive will do.)
- ☐ Choose **Save Performance** from the **File** menu.
- ☐ When the File Requester appears, click on the drive button that corresponds to the drive your disk is in (**DF0:**, **DF1:**, etc.).
- ☐ Click in the **File:** field and type in the name for your performance (end the name with ".Perf").
- ☐ Click **OK** to save your performance.

Well, we've taken a quick tour of Music-X. You've seen the basic landmarks, and started to get some hands-on experience. The next set of chapters give more in-depth technical descriptions of the functions in Music-X. As you read them, please do so with the software running. If you feel you're not ready to dive into the technical chapters yet, you can find more of the step-by-step approach, as used in this chapter, in the Tutorials chapter.

Chapter 5 - File Requester

What are files?31

What are directories?.....32

Requester description 33

How to use the file requester..... 38

Standard filenames 39

Save often and make backups42

File Requester

The next series of chapters are dedicated to explaining the individual Pages in Music-X. This chapter is first in the series because it explains the “file requester”, a window that is available from each Page in Music-X and is central to disk accessing operations. Even if you don’t understand all of Music-X, you’ll at least want to know how to save your songs. You can always come back to them as you learn more about editing, etc.

The file requester appears on the screen whenever you decide to **Save** new data from the computer onto a disk or **Load** previously saved data from disk back into memory. The File requester helps you look into the various directories on a disk to find the file you’re interested in. The same basic file requester window is used by all the pages in Music-X and takes on the colors of each page that uses it; you only need to learn how to use it once. It’s always available from a **File** menu, found in the menu bar when the right mouse button is pushed.

This chapter will first explain some of the terms and concepts that are needed to understand how files are manipulated on the Amiga. It will then give descriptions of the actual features of the file requester in Music-X.

What are files?

The Amiga is an information manipulator. The Amiga uses disks to save information in a permanent form. The Amiga stores information on disk in files. The name “file” is a generic name for any clump of information that is stored on a disk. Files are a way of dividing up information so that it doesn’t get all mixed up together. A file functions as both a container and a label. A file is merely a string of numbers. But those numbers can represent anything from words to pictures to a program like Music-X itself.

When you first run Music-X, the Amiga loads the file that is Music-X and starts it up. Once Music-X is running, it too may create and use its own files. Music-X uses many different kinds of files: some to represent a sequence of notes to be sent to a synthesizer (sequence files), some to represent digitally recorded sounds to be played by the Amiga's sampling chips (sample files), some to hold the settings that tell a synthesizer how to create a certain sound (library files), and there are more.

Each file has its own name. This is called the "filename" (one word). Filenames in the Amiga world can be up to 30 characters long. The characters you can legally use in a filename are all the alphanumeric characters and most punctuation characters, except colon (":"), and virgule ("/"). Amiga filenames are case-insensitive. That is, alphabetic characters are considered to be the same in lower and upper cases. For example, the two filenames, "AmigaFile", and "AMIGAfile", would be considered identical.

What are directories?

A disk typically holds hundreds of different files. To keep all these files organized, a disk is divided into "directories", otherwise known as "drawers." (A drawer is something you put files in. The analogy is with an office file drawer.) A directory separates the files into groups. You see the contents of only one directory at a time. The directory you see is called the "current directory".

There is no limit to the number of files you can put in a drawer, other than the capacity of the disk.

Drawers may be "nested". That is, a drawer may hold not only files, but other drawers as well. (Imagine the analogy of a bag of bags.) An empty formatted disk contains one directory, called the "root directory". When you put new directories on a disk they are actually nested within that root directory.

When speaking of a particular file on disk, it's necessary to describe where that file is by giving it a "pathname". A pathname is divided into three parts: a diskname, a series of directory names, and a filename. A diskname always ends with a colon (":"). Directory names are followed by a virgule, or right-slash ("/"). The filename is last and can be any legal filename. A typical path name would be "*Diskname:directoryname/directoryname/filename*".

To Illustrate the use of nested directories, imagine we have a disk called "Animals". Inside this disk are the two directories "Mammals", and "Amphibians". Inside the "Mammals" directory are the two directories "Primates", and "Marsupials". Now we want to store a file called "Man" in the proper directory. To put man in his place, we use the pathname "Animals:Mammals/Primates/Man".

Requester description

Following is a description of the gadgets that appear in the file requester window. Please look at a file requester while you read this. An easy way to call the file requester is by choosing **Load Performance** from the **File** menu of the Sequencer Page.

Requester Title

The top of the file requester window is titled to show its current function. For example: **Save Performance**, or **Load Filter**, etc. Check to make sure you've chosen the correct function. If not, clicking on the **CANCEL** gadget will get you out of the file requester safely. (You don't want to load when you meant to save, or vice versa.)

OK

Clicking this button tells the software you're finished setting up the parameters, and to go ahead and access the disk. Once you have chosen a drawer and a filename, click on this gadget to execute the save, or load. (Note: Double clicking on a filename is a short cut and is the same as clicking once on the filename and once on **OK**.)

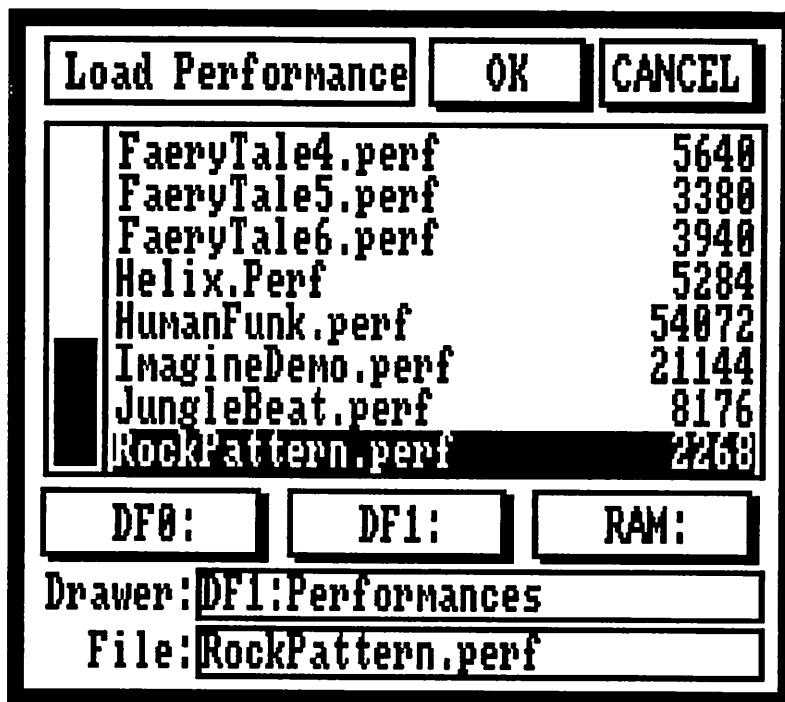


Illustration 5.1 File Requester

CANCEL

This button provides a safe exit from the file requester. Click on this gadget and the window will close without doing anything. You'll be returned to whatever page you were in when you called the requester.

File list

The most prominent feature of the file requester is the file list. The list shows the files and directories in the current directory. The list may be longer than can be shown on the screen. To the left of the list is a "scroll bar". By dragging the bar with the mouse, one causes different portions of the list to come into view in the file list window. Directory names will appear in the list followed by the abbreviation (**dir**) for "directory".

Drive Buttons

Directly below the file list is a cluster of buttons listing the available disk drives. The buttons you see depend on your particular system. When booted, Music-X looks at the hardware attached to your Amiga and finds out how many, and which, disk drives are “mounted” (hooked up and accessible). The file requester is then configured to offer a button for each drive in your system.

A “drive” is the actual hardware box with a motor and record head that reads a disk. It can hold no information of its own. A disk is the separate, small, removable recording medium that holds the actual information. We usually speak of disks by name and drives by number.

The drive buttons are used as a short-cut when typing a path name in the drawer field. Clicking on one first clears the **Drawer:** field (see below), and then enters the name of the drive there. What this actually does is tell the Amiga to look into that drive and use the disk within as the current disk, as if you had typed in the name of the disk itself. In other words, the drive name is substituted for the name of the disk currently in the drive.

Here are descriptions of some drive buttons that may appear, depending on your system. (Customized systems may have other devices than the ones listed here.) Drive names always end with a colon.

DF0:

“Drive Floppy Zero”. This is the internal drive in your A1000, A2000 or A500.

DF1:

“Drive Floppy One”. This is your first extra floppy drive. If you have an A2000, DF1: is mounted internally. If you have an A1000 or A500, your second drive is external.

DF2:, DF3:.

Additional floppy drives will be listed by number.

DH0:

“Drive Hard Zero”. This button will appear if your system includes a hard disk drive. Hard disks are fast and have lots of memory. If you use one, be sure to also make floppy disk backups of all your important Music-X files.

DH1:, DH2:, etc.

Additional hard drives are listed by number.

RAM:

“Random Access Memory”. This is a “virtual” drive in the Amiga’s internal memory. The Amiga sets up a section of its internal memory that it treats as if it were a floppy drive. Anything stored here is lost when you turn off the Amiga. The advantages of RAM: are speed, and availability. RAM: can be used as a temporary storage place, as when collecting favorite performances from various disks, using a single-drive system. If you use the RAM: disk, be sure to save any changes to a real disk before ending your session.

RAD:

Commodore’s “Recoverable Amiga Drive”, is an advanced option available using Workbench 1.3. It too is a “virtual” drive set up in the Amiga’s memory. The difference between RAM: and RAD: is that information stored in RAD: may be recovered after a system error, as long as the Amiga is not turned off. There are also third-party recoverable ramdisks on the market with other names.

Drawer:

Underneath the drive buttons is a gadget titled **Drawer:**. This is a string gadget, meaning you can type in a “string” of characters. String gadgets are also known as “text gadgets” or “text boxes”.

The **Drawer:** gadget is used to specify a drive and directory. By changing the pathname in the text box, one specifies a directory. There are two ways to do this. 1) Click in the box to activate the cursor. Then type the new pathname, and press the RETURN key. 2) Double click on any directory that appears in the file list. The **Drawer:** field will be automatically modified to include the new directory.

Whenever the **Drawer:** field is changed, the file list will change to show the files in the new directory.

If something in the pathname is misspelled or if nested directories are listed in the wrong order, Music-X will give you a message that says “ERROR Couldn’t open the file.” Retype the pathname and try again.

File:

At the bottom of the file requester is the text gadget titled **File:**. This gadget holds the current filename.

If you are loading an old file, here is where you specify which file you want. If you are saving a new file, here is where you first give it a name. If you are resaving an old file, use the same name as when you loaded it. (When saving a file with the same filename as one already in the current directory, the file on disk is erased and the new file is saved in its place. This is how to update files as you work.)

There are three ways to change the **File:** field. 1) Click once in the string area to activate the cursor, then type in the filename and press the RETURN key. 2) Before activating the cursor, tapping an alphabet key on the computer keyboard will cause the first file in the list starting with the letter you typed to be highlighted, and that filename will appear in the **File:** field. 3) Simply click on any filename you see in the list and that filename will appear in the **File:** field.

How to use the file requester

Here's how to quickly locate a file using the file requester.

- ☐ Find the disk that your file is on and put it in a disk drive.
- ☐ Open a file requester (choose **Load...** or **Save...** from a File menu).
- ☐ Click on the drive button for the drive which your disk is in.
- ☐ Wait for all the entries to appear.
- ☐ Scroll through the list and double click on the appropriate directory. (Directory names are followed by the word **(dir).**)
- ☐ Repeat the previous two steps as needed if you have nested directories.
- ☐ Scroll through the list and click on the name of your file (tapping an alpha key is a short cut that searches for and highlights the first file in the list starting with that letter, as if you had clicked it yourself).
- ☐ Or, if you are creating a new file (as when saving), then click in the **File:** field to activate the cursor, and type in the new name.
- ☐ You should now see the **Drawer:** gadget showing the disk and directory(s) that are a path to your file, and the **File:** gadget showing the filename.
- ☐ Click **OK** to Load or Save the file.
- ☐ The requester will then go away, and the mouse pointer will turn into a snooze graphic while disk access is taking place.
- ☐ After disk access is finished, the pointer will return to normal, and you can continue using Music-X.

Standard filenames

Amiga allows you to name directories and files anything you want. A name can be up to 30 characters long including spaces. Most characters are legal except “/” (virgule) and “:” (colon). In most cases, it’s a good idea to keep name lengths down to 25 characters. This is because when Music-X (and other Amiga programs) saves a new file, it also creates an “icon” for that file. Icons are the pictures that you see while in the Amiga Workbench, representing the different files on a disk. An icon is in fact yet another type of file. Every file needs a name. The Workbench expects icon names to be the same name as the file they represent, with the word “.info” appended to the end. Keeping your filenames down to 25 characters in length leaves room for Music-X to append the 5 characters in “.info” so that the Workbench can display icons correctly.

Music-X makes no demands that a filename be anything in particular. However, to make it easier to keep things organized, some standards are suggested here for filenames and directory names when saving data created with Music-X.

Directories are usually named after the Music-X Pages whose files they will hold. Example: “Filters”, “Keymaps”, etc. (Two exceptions: the “Performances” directory is used by the Sequencer page, and the “Modules” directory is used by all pages.)

A filename generally has two parts separated by a period. The first part can be anything descriptive that you choose to write. The second part is called a file extension and is a short word that gives an indication of the file type, often an abbreviation of a Music-X page name. This makes it easy to see what the file may be used for. The file extension is the same for files of a common type. For example, “BigToms.Sample”, “ReallyBigToms!.Sample”, etc.

There are two ways to create your own directories on a disk. One way is with the CLI program and the command **makedir** (see your AmigaDOS documentation).

The other way is from the Workbench. By dragging the icon for the **empty** drawer on the Music-X program disk over the icon for the disk you want the new directory to be created on, and releasing the mouse button, you give the Amiga the command to create an empty directory on the new disk. Once the directory is created, you can rename it using the Workbench command **Rename**.

Following is a list of standard directory names. Directory names are shown followed by the word “(dir)” to distinguish them as directories. (This is supplied by the software and is not actually part of the name.) Listed below each directory name are the standard filenames one might expect to find in that directory. Parts written in *italics* are to be supplied by you. The file extensions and directory names should be used as shown.

Modules (dir)

AnyModuleName

◆ Note: Modules are usually named after the function they perform, with no file extension.

Performances (dir)

AnyPerformanceName.Perf

Sequences (dir)

AnySequenceName.Seq

Filters (dir)

AnyFilterName.Filter

Kmaps (dir)

AnyKeymapName.Kmap

◆ Note: The Amiga Operating system uses a directory named “keymaps” to store files holding the characters used to configure the typewriter keyboard to the alphabets of various countries where Amigas are sold. For this reason we recommend calling Music-X keymaps “Kmaps” to avoid confusion.

Samples (dir)

AnySampleName.Sample

Libraries (dir)

AnyLibraryName.Libr

◆ Note: The Amiga Operating system uses a directory named “libs” to hold files with file extensions of “.library”. For this reason we recommend a slightly different convention.

Protocols (dir)

ManufacturerName (dir)

SynthName.TransferType

◆ Note: Since most manufacturers make more than one synthesizer and most synthesizers support more than one type of data transfer format, we suggest the convention of naming a protocol using the synthesizer model name followed by the type of data that will be transferred (voices, samples, tunings, etc.), and to group these protocols in directories by manufacturer. The manufacturer directories are then nested into the main Protocols directory.

You may wish to ignore the above conventions altogether, and group files together by song, or author, or some other system. In any case, the above examples describe the way the standard Music-X files are organized.

Save often and make backups

Once again, save updates while you work and make backup disks after each session. (Yes, I'm repeating myself). This is something that can't be stressed enough. Accidents can happen. Don't become a casualty.



Chapter 6 - Sequencer Page

Overview	45
Title Bar.	46
Mode Menu.	47
File Menu	50
Options Menu	58
Sync Menu	68
Modules Menu	90
Transport Window.	91
Record Sequence Window	98
Clock Window.	107
Tempo Window	109
Time Signature Window.	111
Sequence List Window.	112
Track Window	117
Store.	118
Edit.	119
Delete.	121
Preview	121

Sequencer Page

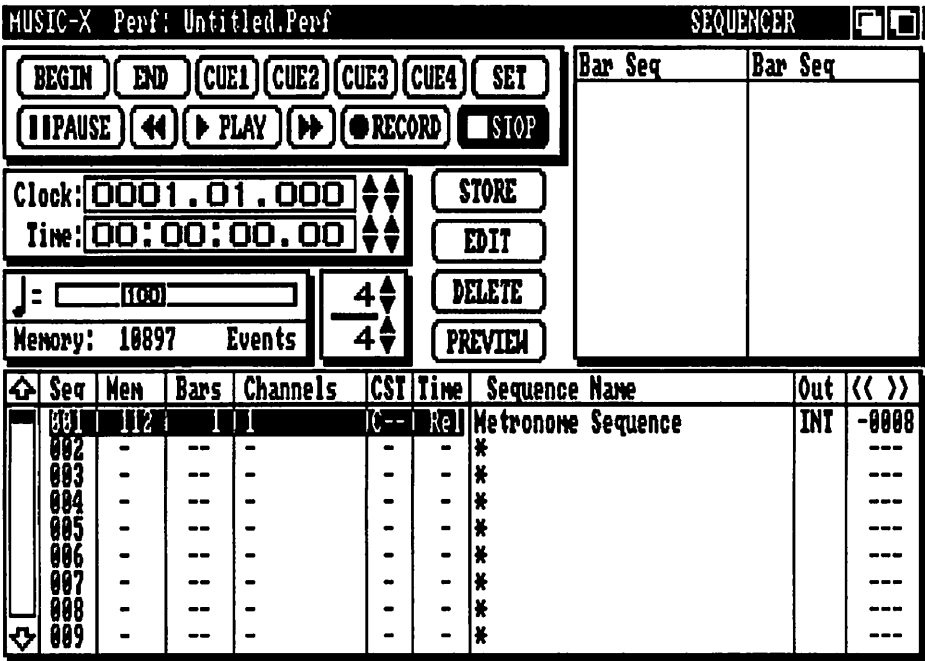


Illustration 6.1

Overview

The Sequencer Page can be considered the primary page in Music-X; Music-X's primary job is to be a MIDI sequencer, and the Sequencer Page contains the features that allow it to do that.

The basic idea behind a MIDI sequencer is to provide a way for a composer to record instrument parts one by one, edit them to satisfaction, and play them back together on MIDI instruments. This is analogous to a multi-track tape recorder, but it goes far beyond the analogy.

There are many different approaches to assembling a piece of music with a sequencer. Some people prefer to record each instrument on its own sequence “top to bottom”, that is, for the whole length of the song. Other people like to overdub all the instruments onto a single sequence the length of one section, and then append various sections to make the song. Music-X was written to accommodate everybody. A sequence may be as long as the whole piece (up to 4096 measures), or as short as one measure, and may contain the data for one instrument or all the instruments. A sequence may also contain events that play other sequences. Up to twenty sequences may be playing at any one time, and there are 250 empty sequences to work with. By combining these attributes, any modality for composing music on a sequencer can be emulated. Some of the various modalities are covered in more detail, in the Advanced Users chapter.

The rest of this chapter describes, in detail, the functions available in the Sequencer Page, roughly in the order that they are found on the screen, left to right, top to bottom.

Title Bar

The Title Bar is found along the top of all pages in Music-X to remind one where one is. “MUSIC-X” is always written at the left end of the Title Bar. At the right end, the title of the current page is written; in the Sequencer Page, this will be “SEQUENCER”. In the Sequencer Page, the name of the current performance will appear written after the abbreviation **Perf:**, in the middle of the title bar. For example, “Perf:MySong.Perf”. This name changes when a new performance is loaded or when the current performance is saved using a new and different name.

The Title Bar becomes the Menu Bar whenever the right mouse button is pressed. The Menu Bar contains the titles of menus that can be accessed by moving the mouse pointer to the menu title as you continue to hold the right mouse button down.

[see Sequencer Page / File Menu / Load Performance]

[see Sequencer Page / File Menu / Save Performance]

Mode Menu

The Mode menu is found in each of the four main pages in Music-X. By choosing the four pages listed in the Mode menu, one moves about in Music-X. Additional pages are accessible only from within the main pages, using other menus or screen controls.

Below are descriptions of the menu items found in the Mode menu, including short descriptions of the main pages and any ancillary pages accessible from each of them.

About Music-X

This menu item brings up a window that shows the Author of Music-X, and the author and title of the current performance in memory. If you have previously set your song's title and author, they will appear in this window. Clicking the left mouse button, or pressing any key, closes this window.

[See Sequencer Page / File Menu / Set Author Name]

Suspend

(RightAmiga-“.” from the keyboard)

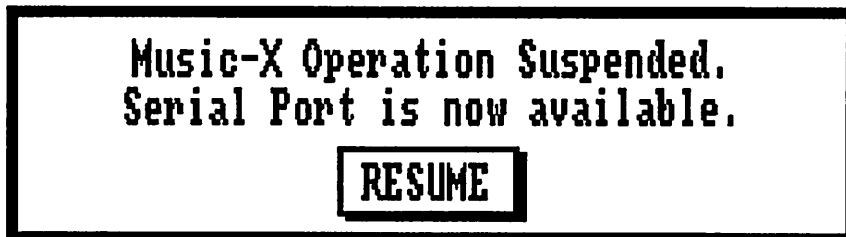


Illustration 6.2: Suspend Prompt.

MIDI uses the serial port. To maintain data integrity, Music-X takes over the serial port and locks out other programs. If you need to temporarily use the serial port for another application (for instance, a telecommunications program), you can suspend Music-X by choosing this menu item. As long as this window is showing, Music-X is suspended. When you're finished, click **RESUME** or press the RETURN key to resume the Music-X session.

Sequencer

(RightAmiga-“S” from the keyboard)

Choosing this menu item will get you back to the Sequencer Page from any of the other three main pages where the Mode menu is available (Filters Page, Amiga Samples Page, Librarian Page). The Sequencer Page is used to record, edit, and play musical sequences. The Bar Editor and Event Editor Pages are accessible from this page as well.

Set Filters

(RightAmiga-“F” from the keyboard)

This is how you get to the Filters Page, from which you may also access the Keymap Editor Page. These pages are used to reroute or reinterpret incoming MIDI data.

Amiga Samples

(RightAmiga-“A” from the keyboard)

This is how you get to the Amiga Samples Page, where you load, manipulate, and save sounds for the internal Amiga voices.

Librarian

(RightAmiga-“L” from the keyboard)

This is how you get to the Librarian Page. From there you can send and receive system-exclusive patch data for any synthesizers or MIDI devices that you have connected to your system.

Quit

(RightAmiga-“Q” from the keyboard)

Choosing this menu item will quit Music-X and return to the Amiga Workbench. Be sure to save your music before quitting if needed.

[See Sequencer Page / File Menu / Save Performance]

A safety prompt will appear to prevent you from leaving the program by accident. If you change your mind, click **NO** and you'll be safely returned to Music-X. If you indeed want to quit, click **YES** or press the RETURN key. Music-X will graciously return the memory to your Amiga's Operating System.

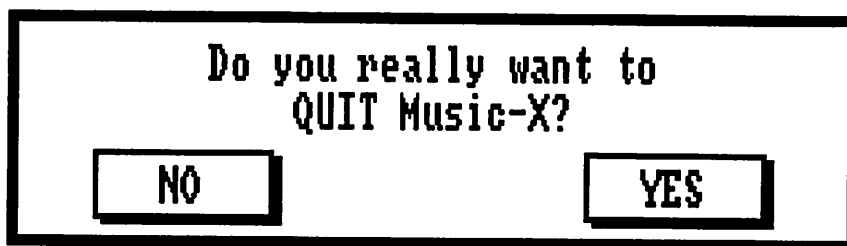


Illustration 6.3: Quit Safety Prompt

File Menu

A File menu is found in every page in Music-X. Each page can load and save disk files relevant to its particular function. The Sequencer Page is unique, however, in that it can also Load, Save, and Merge Performance files. A single Performance file can contain all the files as would be saved by each page individually. These component files in a Performance file are called “subfiles”. A list of the various subfile types is found below under the description of the **Save Performance** function.

When you boot Music-X by double clicking on its icon in the Workbench, Music-X looks on the Music-X Program disk for a performance file called “Default.Perf”. It then loads this performance file and finishes booting itself. By rewriting the default performance file, you can have Music-X boot in any configuration you want.

You can also boot Music-X by double clicking on the icon of one of its performance files. In this case, the chosen performance file is loaded instead of the default performance file.

Following are descriptions of the items found in the Sequencer Page / File Menu.

Merge Performance

Choosing this menu item loads a previously saved performance file into memory from disk, without first erasing the sequences currently in memory. Sequences being merged in from the new performance file will overwrite only those current sequences occupying the list positions that they use. (This is similar to the “Merge to Front” function found commonly in paint programs.) Any old sequences not rewritten by the loading of the new performance file will still be in memory after the merge. Any other subfiles (Libraries, Keymaps, etc.) in the new performance file will be loaded during the merge as well.

If you need more control over the transporting of sequences between performance files, then Load or Save the sequences individually.

[see Sequencer Page / File Menu / Load Sequence (below)]

[see Bar Editor / File Menu / Load]

[see Bar Editor / File Menu / Save]

Load Performance

(RightAmiga-“P” from the keyboard)

This menu item is used to Load a previously saved performance file into memory from disk. Any subfiles, loaded with the performance file, overwrite the relevant data in Music-X. That is, if Keymap subfiles are present in the Performance file, they overwrite the current Keymaps in Music-X. Or, if Filter subfiles are present in the Performance file, they overwrite the current Filters in Music-X. Conversely, if a subfile is not present in the performance file, the relevant data remains unaffected. The two exceptions are with Sequence subfiles and Library subfiles. Unlike **Merge Performance**, **Load Performance** erases all the old sequences and libraries in memory before loading the new performance file. Each time you load a new performance file, you should go to Librarian Page and check if any libraries were loaded with the performance, and if so, send them to the synthesizers in your system.

[See “Save Performance” below for descriptions of the subfile types.]

After you choose **Load Performance**, the file requester window will appear. Use this requester to locate the performance file you wish to load.

[see File Requester chapter]

If the Amiga runs out of memory loading a big performance file, Music-X will put up a “Not Enough Memory” prompt. If this happens, close any other programs currently in the Amiga’s memory, and try again.

◆ Note: It’s possible to load music written on other music programs by first converting their files to Music-X performance files. Programs are provided on the Music-X Utilities disk to do this.

[see Advanced Users / File Conversions]

Save Performance

(RightAmiga-“V” from the keyboard)

A performance file is like a “snapshot” of the status of all the various Music-X pages. **Save Performance** lets you save all the parameters in one composite file called a performance file.

◆ Note: If you want to show off multitasking, try saving a performance while it’s playing back.

Choosing **Save Performance** first calls the Subfile window, where you set which subfiles are to be included in the performance file. A list of subfile types is given with a switch for each. The default action is to save everything; all the switches are on and highlighted in light blue.

◆ Note: If a subfile is NOT included in a performance-file save, then it will not be reloaded later and will not overwrite any related parameters at that time.

After setting the subfile switches, click **OK** to close the Subfile window and continue to the **Save Performance** File Requester, or press “O” for (O)K or “Y” for (Y)es. Or, if you decide against saving the performance file after all, click on **CANCEL**, or press “C” for (C)ANCEL or “N” for (N)o.

Once you reach the File Requester, use it to locate the correct disk and directory. Then type in the name of your new performance file in the **File:** text box, and click **OK** to save the performance. Music-X will also create and save an icon for the performance file in the same directory as the performance file. Performance file icons look like music cassettes. The Default.perf icon in the Music-X program disk is an example of a performance icon.

[see File Requester chapter]

◆ Note: Remember to back up your data disk at the end of the sequencing session!

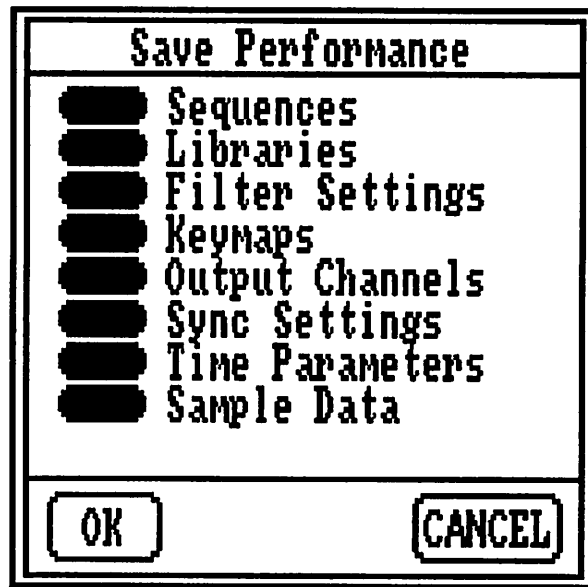


Illustration 6.4: Subfile switches

Following are descriptions of the subfile switches.

Sequences

Any non-empty sequences are saved with the performance file if this switch is on. You could conceivably choose not to save sequences, if you were creating some sort of blank, template-type performance file with all the other parameters set up in a particular way; later, when reloading the template, any sequences in memory would not be erased, but the rest of the parameters would be reset.

◆ Note: Individual Sequence files may be loaded and saved from the Editor Pages.

[see Bar Editor / File Menu / Load]

[see Bar Editor / File Menu / Save]

The current Song Title and Author, as set with the **Set Author Name** option in the File menu, are also saved in the performance file when the **Sequences** switch is on.

[see Sequencer Page / File Menu / Set Author Name]

Libraries

This subfile switch saves all patch libraries currently in memory. (Up to 10 libraries can be resident in memory at once.) Later, when you load the performance file, the patches will be available to send to the synths from the Librarian Page. This allows reproduction of the specific external sounds you were using when you first created and saved your performance.

◆ Note: Individual Library files may be loaded and saved from the Librarian Page.

[see Librarian Page / File Menu / Load]

[see Librarian Page / File Menu / Save]

Filter Settings

This subfile switch causes the current settings for all 16 filters in the Filters Page to be saved with the performance file. Settings include the **Remap** and **ENABLE** switch states, and the continuous slider positions. Keymaps are not included; they have their own switch (see below).

◆ Note: Individual Filter files may be loaded and saved from the Filters Page.

[see Filters Page / File Menu / Load]

[see Filters Page / File Menu / Save]

Keymaps

This subfile switch causes the four Keymaps, created in the Keymap Editor Page and used by the Filters Page, to be saved with the performance file.

◆ Note: Individual Keymap files may be loaded and saved from the Keymap Editor Page.

[see Keymap Editor / File Menu / Load]

[see Keymap Editor / File Menu / Save]

Output Channels

This switch saves the settings from the Channelizer. This is handy if you have standard channel assignments for your MIDI instruments, but your performance contains different channel data. Configure the Output Channels to match your equipment's channel assignments and save the settings with your performance file.

[see Sequencer Page / Options Menu / Output Channels]

Sync Settings

When this switch is on, all the settings of the Sequencer Page / Sync Menu are saved in your performance file. This is handy if you use a drum machine or are doing film or video scoring. At reload, the old sync settings will be duplicated eliminating the need to manually reconfigure your system.

The Audio Filter setting, as set in the Samples Page, is also saved in the performance file when the **Sync Settings** switch is on.

[see Samples Page / Options Menu / Audio Filter]

Time Parameters

This switch saves the current master Tempo setting, the current master Time Signature setting, and the locations stored in the **CUE** buttons.

[see Sequencer Page / Tempo Window]

[see Sequencer Page / Time Signature Window]

[see Sequencer Page / Tape Transport Window / CUE Buttons]

Sample Data

When this switch is on, any samples currently in memory will be saved along with your performance file.

Sample data can use lots of disk memory. If you have a standard set of samples that you always use, then there's no need to keep resaving them with your performance files. You may wish instead to create a "sample template" performance file by first loading your favorite samples individually using the Samples Page, and then saving a performance file with all of the subfile switches turned off except **Sample Data**. Then you'd only need to load this one performance file once, before playing any of the songs that use those samples.

◆ Note: Individual Sample files may be loaded and saved from the Samples Page.

[see Samples Page / File Menu / Load]

[see Samples Page / File Menu / Save]

Load Sequence

This item in the File menu is used to Load a sequence file from disk into the current sequence (highlighted in the sequence list), overwriting its previous contents.

After you choose **Load Sequence**, the File Requester will appear. Use it to locate the file you want and click **OK** to load the sequence.

[see File Requester chapter]

◆ **Note: Save Sequence** has not been made available from the Sequencer Page / File Menu, to prevent the accidental saving of a single sequence when the intention was to save all sequences in a Performance file. Individual sequences can be Loaded and Saved from within the Bar Editor or Event Editor.

[see Bar Editor / File Menu / Load]

[see Bar Editor / File Menu / Save]

Set Author Name

Here is where you set the Title and Author of the current performance in memory. These settings will appear in the **About Music-X** box, and are automatically saved with any performance file.

[see Sequencer Page / Mode Menu / About Music-X]

	Song Title: Peanut Butter Jamboree
	Author: Sticky Vicky

Illustration 6.5: Set Author Name Window

When you choose this menu item, the above window appears showing the current settings. Here is how to change those settings.

- ☐ Click in the **Song Title** text box, then use the Amiga's keyboard to type in the name.
- ☐ Click in the **Author** text box, then use the Amiga's keyboard to type in the name.
- ☐ When done, click **OK**.

Options Menu

The Options menu provides extra functions for the Sequencer Page. The Output Channelizer is accessed from here as well as some global sequence editing functions like **Merge** and **Extract**.

Following are descriptions of the items available in the Sequencer Page / Options Menu.

Output Channels...

This menu item calls the **CHANNELIZER** window.

What do you do if you load up a friend's song, in which all the drum parts were recorded on MIDI channel 2, but uh-oh... your drum module **ONLY** receives on channel 10 !?

You can either go down the list and edit the MIDI channels in all the sequences that contain drum parts, one by one, or, you can redirect your output channels using the Channelizer.

The Channelizer globally translates all MIDI channel data during playback only. For each original MIDI channel there is a translated MIDI channel. The original data is left intact. It's safe to experiment.

The **From** column shows all the possible original MIDI channels (the channels your data is on). The **To** column shows the channels you'd like each of these original channels to be translated to. The **Set** column is clicked on to change the **To** column; it's merely a list of the numbers 1-16 provided for speed.

Example: To switch those drum channels, click on "02" in the **From** column. Click on "10" in the **Set** column. Notice the **To** column now shows "10". Now all the

CHANNELIZER		
From	To	Set
01	--> 01	01
02	--> 10	02
03	--> 03	03
04	--> 04	04
05	--> 05	05
06	--> 06	06
07	--> 07	07
08	--> 08	08
09	--> 09	09
10	--> 10	10
11	--> 11	11
12	--> 12	12
13	--> 13	13
14	--> 14	14
15	--> 15	15
16	--> 16	16

Illustration 6.6: Output Channelizer

drum parts stored in sequences on MIDI channel 2 will actually be played on MIDI channel 10.

◆ Note: Try switching the Channelizer while Music-X is playing. This works, and might give you a better indication of what it's doing.

What's really going on: All Music-X data is recorded and stored with the MIDI-channel "nybble" intact. Data is passed through the channelizer on its way out. The channelizer uses the channel nybble to index a lookup table and substitutes the number found there for the original channel. This window allows you to edit the table.

◆ Note: Only notes played back from Music-X sequences are rechannelized. The channelizer does not affect incoming notes played on the mother keyboard. Use the Filters Page to remap incoming MIDI data.

◆ Note: The Channelizer settings can be saved with your performance file.

[see Sequencer Page / File Menu / Save Performance]

Set Punch-In

(Function key F1 from the keyboard)

Choosing this menu item sets the automatic Punch-In point to whatever the current clock setting is. First set the musical clock to the desired Measure, Beat and Subclock. Then choose this menu item. This can also be done "on the fly" when in **PLAY** or **RECORD** mode by tapping the F1 key. Punch-In is explained in more detail in the section on recording.

[see Sequencer Page / Transport Window / RECORD]

Set Punch-Out

(Function key F2 from the keyboard)

Choosing this menu item sets the automatic Punch-Out point to whatever the current clock setting is. First set the musical clock to the desired Measure, Beat, and Subclock. Then choose this menu item. This can also be done “on the fly” when in **PLAY** or **RECORD** mode by tapping the F2 key.

Copy Sequence

(RightAmiga-“C” from the keyboard)

You may wish to keep several versions of a sequence while perfecting it. **Copy Sequence** can be used to create the extra versions.

Choosing this menu item will copy all the data from the current sequence (source) to any other sequence in the list (destination). The source sequence remains unaffected. Anything previously in the destination sequence is, of course, lost.

The source sequence is always the one highlighted before you choose **Copy Sequence**.

Choosing **Copy Sequence** calls the **SELECT SEQUENCE TO COPY TO** window. Use this window to define the destination sequence by scrolling through the sequence list with the CursorUp and CursorDown keys or by dragging the scroll bar, and clicking on an empty or expendable sequence. Then click **OK**, or press “O” or “Y” to effect the copy. Or, click **CANCEL** (or press “C” for (C)ANCEL or “N” for (N)o) to safely exit this function with no changes made.

The **Copy Sequence** option and the next two options in the Options menu, **Merge Sequence** and **Extract Sequence**, use the “Edit Buffer” as a temporary working area. The Edit Buffer is also used to record and edit sequences. If the Edit Buffer is not empty when you choose any of these options, a safety prompt will appear to prevent accidental erasure of sequence data.



Illustration 6.7: Buffer unsaved warning

The prompt has three choices. Clicking **CANCEL** will abort whichever option you were about to use and return you to the sequencer page. Clicking **DISCARD** will lose the current contents of the Edit Buffer and continue with the option as intended. Clicking **STORE** will open a sequence selector window titled **STORE TO WHERE?**, offering a list of the 250 possible sequences in memory. Use the window to choose a sequence and click **OK** to store the current contents of the Edit Buffer to that sequence, clearing the Edit Buffer for use. After the Edit Buffer is cleared, you can continue with the originally chosen option.

Merge Sequence

(RightAmiga-“M” from the keyboard)

Use this menu item to merge the data from the current sequence (source) into any other sequence in the list (destination). The source sequence remains unaffected (unless merging to itself). The destination sequence will contain all its old data plus the data from the source sequence. Merging a sequence to itself will create a MIDI doubling of all the notes in the sequence.

The source sequence is always the one highlighted before you choose **Merge Sequence**.

Choosing **Merge Sequence** calls the **SELECT SEQUENCE TO MERGE TO** window, showing a sequence list. Scroll through the list with the CursorUp and CursorDown keys or by dragging the scroll bar with the mouse. Choose the destination sequence by clicking on an empty or expendable sequence to highlight it. Then click **OK** to effect the merge. Or click **CANCEL** (or press “C” for (C)ANCEL or “N” for (N)o) to safely exit this function with no changes made.

Once the merge is initiated Music-X will temporarily ignore the mouse, to help speed up the merging process. The amount of time for the merge to complete depends on the sizes of the sequences being merged. When Music-X is done merging, it will put up a prompt saying “Merge Finished”. Click on the **OK** button to close the prompt and return to the Sequencer Page.

Extract Sequence

Think of this as “Extract a new sequence from an old sequence”. You essentially extract a copy of just the good stuff without affecting the original.

What Extract does is selectively copy into the Edit Buffer chosen classes of events from the contents of the current sequence without altering that sequence. You must then manually store the buffer containing the extracted events to a new sequence (using the Sequencer Page / STORE button).

The source sequence is the one highlighted before you call **Extract Sequence**. The destination sequence is actually the Edit Buffer until you store it to a spare sequence.

There are three basic event classes that can be Extracted: Channel events, System Exclusive data, and Music-X events.

Channel events are those MIDI events that control instruments on a particular MIDI channel: Notes, Channel After Touch (CAT), Polyphonic After Touch (PAT), Control Changes (CTL), Program Changes (PGM), and Pitch Bend (PBEN).

System Exclusive data varies with different manufacturers and is only understood by the instrument it's intended for.

Music-X events are events that internally control the sequencer in some way: Play Sequence (PSEQ), Mute Sequence (MSEQ), Solo Sequence (SSEQ), Mute Track (MTRK), Solo Track (STRK), Set Repeats (REPT), Set End (END), Change Keymap (KMAP), Tempo Change (TEMPO), Time Signature Change (TSIG), and Stop Sequencer (STOP).

◆ Note: Event types are described in detail in the Editor chapters.

[see Bar Editor / Event types]
[see Event Editor / Event types]

After you choose **Extract Sequence** a window appears titled **Extract Which Channels?**. All the channels are listed, as well as the two classes of non-channel events: Music-X events, and System Exclusive events. Click on the channel numbers or the other event types (they will appear highlighted in white) to choose which will be copied into the Edit Buffer (clicking them again turns them off again).

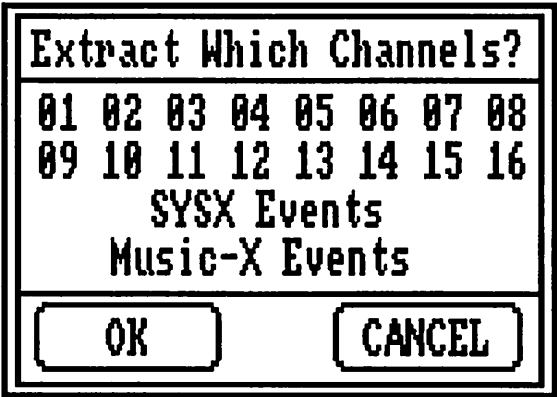


Illustration 6.8: Extract Window

If any of the channel numbers are switched on, then Channel events on the chosen channel(s) will be copied into the Edit Buffer.

If the **SYSX Events** switch is on then any System Exclusive data in the source sequence is copied into the Edit Buffer.

If the **Music-X Events** switch is on, then any Music-X events present in the source sequence will be copied into the Edit Buffer.

How to Extract:

Say we had a composite sequence called “Bass and Drums” containing two instruments. The drums are on MIDI channel 1 and the bass is on MIDI channel 2. Here’s how we’d separate the instruments into their own sequences:

- ☐ From the Sequencer Page, find the “Bass and Drums” sequence in the sequence list and highlight it. This is our source sequence for the Extract function.
- ☐ Choose **Extract Sequence** from the Options menu. The **Extract Which Channels?** window will appear.
- ☐ When the requester appears, choose to extract data on MIDI channel 1 by clicking “01”. (It should become highlighted). Make sure no other MIDI channels are highlighted; we only want the drum data.
- ☐ Click **OK** to start the extraction.

◆ **Note:** If the Edit Buffer was not empty when you clicked **OK** a safety prompt will appear so you can decide what to do with the data previously in the buffer. Clicking **DISCARD** in the safety prompt will lose what was in the buffer and let you proceed.

- ☐ A prompt will appear when the extraction is finished. Click **OK** to proceed.

- ☐ We now have a copy of the drums in the Edit Buffer. Choose an empty sequence by highlighting it in the sequence list and click **STORE** to write the Edit Buffer into the empty sequence. The new sequence now contains the drum part. Title it “Drums alone”, or something, by clicking in the **Sequence Name** field and typing in the name.

We’ve extracted a copy of just the drums and stored it to a new location. Now we’ll do the same thing for the bass line:

- ☐ Select the “Bass and Drums” sequence again and call **Extract Sequence**.
- ☐ This time deselect MIDI channel 1 (drums), by clicking “01” to turn it back off, and select MIDI channel 2 (bass) by clicking on “02”. Then click **OK**. And again **OK** at the “Extract Finished” prompt.
- ☐ We now have the bass in the Edit Buffer.
- ☐ **STORE** the Edit Buffer to a new sequence and rename it to something like “Bass alone”.

Now that we are finished Extracting, we can **DELETE** the original composite sequence or keep it as a backup.

If we were really slick, we could have written the Edit Buffer containing the bass part back to the original “Bass and Drums” sequence, effectively doing a **DELETE** and a **STORE** in one step.

Erase All Sequences

Choosing this menu item erases all sequences currently in memory, but nothing else. Cue points, Tempo, Sync settings, and other parameters are all left in their current state. This can be used to clear out sequence space when starting a new piece

of music. A safety prompt will then appear to prevent inadvertent loss. Click on **YES**, or press the RETURN key to erase the sequences. Click on **NO** to safely return to Music-X without erasing the sequences.

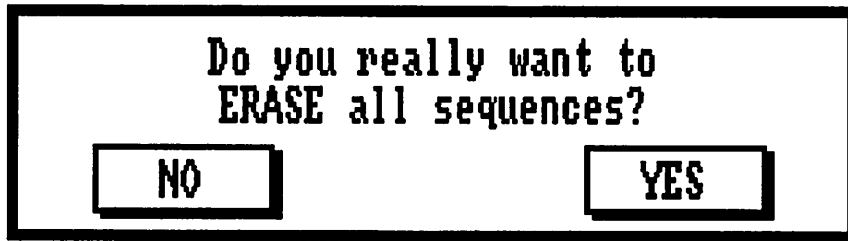


Illustration 6.9: Erase Safety Prompt

Use Zero Origin

This causes the master clock's measure and beat numbers to start at 0000.00.000 instead of 0001.01.000, in case you prefer thinking that way or need to do some frame calculations, etc. All pages are affected. This option is active if the menu item is check marked.

[see Sequencer Page / Clock Window]

Workbench

This option is part of a memory saving technique. In the Amiga, all the graphics must be stored in special memory called chip memory, if they are to be displayed. Different applications (programs) reserve sections of chip memory to use for their graphic "screens". The Amiga's Workbench uses a screen and takes up some chip memory. If Music-X boots up into a system with little free memory, it may turn off the Workbench screen to free up some of the chip memory. This does not mean that the Workbench is erased or stops running, it just means the screen is not displayed to save chip memory. Music-X does this so it will be able to display its own menus, windows, requesters, and prompts.

While checked, this menu item means that the Workbench screen is turned on. One can see and move between the Music-X and Workbench screens by clicking the front-to-back gadgets or by pulling one down to reveal the other.

[see Tutorials / Amiga Basics]

When this menu item is not checked, it means the Workbench screen has been removed from chip memory and is turned off. The Workbench can be turned back on by selecting this menu item.

Sync Menu

The word synchronous comes from the Greek words: *sun* for “same”, and *khronos* for “time”. The Sync menu governs the way Music-X synchronizes with, or operates at the “same time” as, other time keeping devices.

Settings made in this menu can be saved and reloaded with any performance file (including the Default performance), making the reconstruction of sync setups automatic.

◆ Note: If you consistently use the same equipment set up, you might want to resave the “Default.perf” file with your particular Sync menu settings.

[see Sequencer Page / File Menu / Save Performance]

The need for synchronization

Music is a time dependent art form. The applications of electronic music often involve the use of several time keeping devices at once (drum machines, sequencers, audio tape recorders, film or video equipment, etc.). It’s rare that any two unconnected time keepers can agree with each other without eventually drifting apart. Even if they could, it would be difficult to independently start them at exactly the same time. It’s therefore necessary to synchronize them somehow.

This is usually done by making one device the leader, or master, and using it to transmit start, stop, location, and timing information that the others receive and follow. That way, even if the leader is wrong, they all move together. A common method is to code “messages” into binary numbers and transmit them electronically as very fast two-state voltage fluctuations. MIDI and SMPTE are two examples of this method.

◆ **Note:** Music-X has the ability to operate in synchrony with a number of external time sources (be a follower). It can also transmit MIDI Clock and Song Position Pointer messages to synchronize drum machines and sequencers (be a leader). In some cases it can be a leader and a follower simultaneously.

Two types of time

Synchronization messages can be divided into two types, those that transfer Relative Time information and those that transfer Absolute Time information.

Relative Time is described in musical terms like “Measures”, and “Beats”. When you say “the second measure of the bridge”, you are speaking of Relative Time. Two examples of Relative Time synchronization are MIDI Song Position Pointer and MIDI Clocks (more on those later).

Absolute Time is described in Hours, Minutes, and Seconds. When you say, “The gig starts at seven thirty”, you are speaking of Absolute Time. An example of Absolute Time synchronization is SMPTE Time Code (more on that too).

Music-X has two types of sequences, those that are tempo dependent (Relative sequences), and those that move with the clock (Absolute sequences). Music-X also has two internal clocks, one for each type of sequence (Relative and Absolute).

The rest of this chapter contains individual descriptions of the items found in the Sync menu.

Master Clock

The **Master Clock** setting determines which source will be monitored to keep Music-X moving through time. Music-X actually always runs on its own internal clock. As timing messages are received from the current Master Clock source, Music-X's internal clock is reset to match them. This provides as fine an internal resolution as possible by allowing Music-X to play between incoming messages, while making sure that, as messages come in, Music-X keeps in step with them.

Two internal and three external time sources are listed in a pop-out menu. The five choices are mutually exclusive. The checked item shows the current **Master Clock** setting. The choices are described below.

Internal

When **Internal** is selected, Music-X uses one of the Amiga's timer chips as the Master Clock source.

The Relative clock is updated in increments of 1/192 quarter note, at a rate dependent on the current tempo. The Absolute clock is updated through a conversion routine.

Example: Playing Music-X with no other sync equipment.

☐ Set **Master Clock** to **Internal**

☐ Click **PLAY** to start Music-X

MIDI Clocks

Read this menu item as "MIDI Clock Messages". When this Master Clock setting is chosen, Music-X synchronizes its internal clock to incoming MIDI Clock messages while in **PLAY** or **RECORD** modes.

What are MIDI Clock messages?

MIDI Clocks are MIDI messages transmitted by a drum machine or sequencer that say to the receiver, "Advance your musical clock by 1/24 quarter note." Each time the transmitter advances its own internal clock by 1/24 quarter note, it sends out a MIDI Clock message. Each time the receiver gets one of these messages, it advances its clock as well. This means that whatever the tempo of the transmitting device, the receiver will follow it; the transmitting device controls the tempo of the music.

MIDI Clocks is used as the Master Clock setting in situations where an external drum machine or sequencer will be the time keeper, and you wish Music-X to synchronize with it.

♦ Note: In most situations where a drum machine will be used, it is better to let Music-X be the master, and have the drum machine follow Music-X. This allows Music-X to be in charge of tempo changes, and to play Absolute sequences correctly. (MIDI clocks, being a Relative Time system, cannot transmit Absolute Time information.) However, the **MIDI Clocks** menu option has been provided for other situations.

[see Sequencer Page / Sync Menu / MIDI Sync Out]

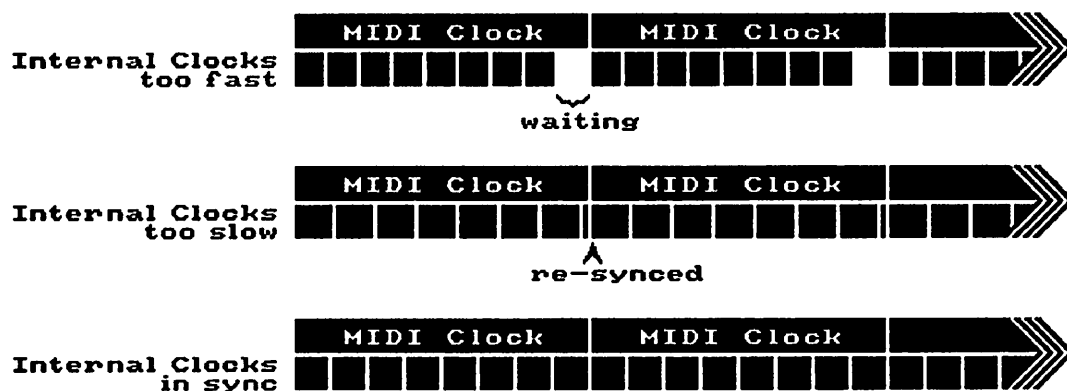


Illustration 6.10: Improving the resolution of MIDI Clocks.

MIDI Clocks have a resolution of 1/24 quarter note. Music-X has a resolution of 1/192 quarter note. That means that Music-X has eight times the resolution of MIDI clocks. It seems a shame to waste all that resolution. However, by matching Music-X's tempo changes to the drum machine's, one can effectively increase the resolution and accuracy of MIDI Clocks.

Example: Using a drum machine to drive Music-X.

- ☐ The drum machine's MIDI OUT must be connected to the Amiga's MIDI IN and the drum machine must be able to transmit MIDI Clocks.
- ☐ Set **Master Clock** to **MIDI Clocks**.
- ☐ Set **MIDI Start & Stop** (also found in the sync menu) to **Receive**.
- ☐ Manually start the drum machine.
- ☐ The drum machine sends a MIDI Start message.
- ☐ Music-X receives the MIDI Start message and locates its musical clock to the beginning of the song.
- ☐ The drum machine sends the first MIDI Clock message and begins playing.
- ☐ Music-X receives the first MIDI Clock message and begins playing.
- ☐ Music-X continues to play the next 8 of its own clocks (the equivalent of 1/24 quarter note) at the current Music-X tempo.
- ☐ If Music-X finishes playing its 8 clocks before receiving the next MIDI Clock message, then it stops playing and waits.

- ☐ The drum machine, advancing at its own tempo, sends the next MIDI Clock message.
- ☐ Music-X receives the next MIDI Clock message, advances its musical clock to the next 1/24 quarter note, and continues to play. If the MIDI Clock is received before Music-X expected it, then Music-X jumps ahead anyway to keep in step. (See Illustration 6.10)
- ☐ Music-X matches the tempo of the drum machine and provides some additional resolution as well. This process continues until the end of the song.

[see Sequencer Page / Sync Menu / MIDI Start & Stop]

[see Sequencer Page / Tempo Window]

◆ **Note:** Most of the newer drum machines transmit MIDI Clocks even while stopped. This means that the sequencer can play while syncing to MIDI Clocks, even without the drum machine playing.

MIDI Clocks can also provide a way of moving songs written on another computer-sequencer over to the Amiga:

- ☐ The source computer must be able to transmit MIDI Clocks and its MIDI OUT must be connected to the Amiga's MIDI IN.
- ☐ Set **Master Clock** to **MIDI Clocks**.
- ☐ Set the two sequencers' tempo sliders to the same value.
- ☐ Put Music-X in **RECORD**.
- ☐ Play the other sequencer.

- ❑ All the MIDI data played by the source will be recorded by Music-X in time with the MIDI Clock messages.

[see Sequencer Page / Transport Window / RECORD]

[see Sequencer Page / STORE]

Video Clock

When **Video Clock** is chosen as the **Master Clock** setting, Music-X uses the Amiga's internal video clock to drive the Absolute clock.

There is a chip in the Amiga that generates the timing frequency used to drive the monitor display's Vertical Blanking Interval (VBI). The VBI is the dark strip between frames in a television signal. The Amiga's VBI frequency is just slightly under 60 frames per second. When using **Video Clock** as the Master Clock, the Absolute clock is updated by 1/2 frame at every VBI. Since the video chip frequency is the same as the National Television Systems Committee (NTSC) standard frame rate for television, the **Video Clock** setting can be used to approximate Non-Drop Frame, or Drop Frame SMPTE in cases where you want to test the total elapsed time of the piece for accuracy against the clock.

MIDI Time Code

When MIDI Time Code is chosen as the Master Clock setting, Music-X synchronizes its Absolute clock to incoming MIDI Time Code messages.

What is MIDI Time Code?

MIDI Time Code (MTC) is a type of MIDI message that describes elapsed time in Hours, Minutes, Seconds, and Quarter Frames. This makes it similar to SMPTE Time Code, with the advantage that MTC can be received via normal MIDI cables; SMPTE Reader hardware is not necessary.

MTC has four formats: 24 FPS (Frames Per Second), 25 FPS, Non-Drop Frame, and Drop Frame. When synchronizing Music-X to MTC, it's necessary to set Music-X to the same format using the **Time Code** parameter.

[see Sequencer Page / Sync Menu / RECORD]

Example: Using a SMPTE to MIDI Time Code converter box to sync Music-X to 24 track tape.

- ☐ "Stripe" the 23rd track of your tape with SMPTE Time Code, using a conventional SMPTE unit.
- ☐ Connect the output from the 23rd track to the SMPTE input of the converter box, and connect a MIDI cable from the converter's MIDI OUT to the Amiga's MIDI IN.
- ☐ Set **Master Clock** to **Midi Time Code**.
- ☐ Set the **Time Code** to the frame rate that you used when you striped the tape. (**Drop Frame** is the usual choice.)
- ☐ Set the **Time Code Offset** option if needed. (This is explained later.)
- ☐ Click **PLAY** to ready Music-X.
- ☐ Play the tape.
- ☐ The tape machine will send SMPTE Time Code to the converter box and the converter box will send MIDI Time Code to the Amiga, and Music-X.

- ☐ When Music-X receives the first Time Code message it figures out if the time specified by the message is more than 1 measure away from it's own current location. If so, it relocates to the spot dictated by SMPTE.
- ☐ When Music-X has finished locating, it will start to play along with the tape.
- ☐ As you stop, rewind, fast forward, and play the 24 track tape machine, Music-X will automatically **STOP**, relocate, and **PLAY** to follow the movements of the tape.

SMPTE Reader

When **SMPTE Reader** is chosen as the Master Clock source, Music-X synchronizes its Absolute clock to incoming SMPTE Time Code messages received through an external hardware SMPTE Reader box.

What is SMPTE?

SMPTE (pronounced "Sim-tee"), or more properly SMPTE/EBU, is an acronym for "Society of Motion Picture and Television Engineers / European Broadcasting Union". In 1969 these two institutions agreed upon the standard time code formats used today. SMPTE Time Code is a method of describing time in terms of Hours, Minutes, Seconds, and Frame numbers, for use in motion pictures and television. SMPTE Time Code has four formats, for the four common frame rates: 24 FPS (Frames Per Second), 25 FPS, Non-Drop Frame, and Drop Frame.

Example: To sync Music-X to 24 track tape, using SMPTE.

- ☐ "Stripe" the 23rd track of your tape with SMPTE Time Code, using a conventional SMPTE unit.
- ☐ Connect a SMPTE Reader to the Amiga and feed it the output from the 23rd track.
- ☐ Set **Master Clock** to **SMPTE Reader**.

- ☐ Set **Time Code** to the frame rate that you used when you striped the tape. (**Drop Frame** is the usual choice.)
- ☐ Set the **Time Code Offset** option if needed.
- ☐ Click **PLAY** to ready Music-X.
- ☐ Play the tape recorder.
- ☐ The tape recorder will send SMPTE Time Code to the SMPTE Reader and the SMPTE reader will send it to the Amiga, and Music-X.
- ☐ When Music-X receives the first Time Code message it figures out if the time specified by the message is more than 1 measure away from it's own current location. If so, it relocates to the spot dictated by SMPTE.
- ☐ When Music-X has finished locating, it will start to play along with the tape.
- ☐ As you stop, rewind, fast forward, and play the 24 track tape machine, Music-X will automatically **STOP**, relocate, and **PLAY** to follow the movements of the tape.

◆ **Note:** There are two schools when it comes to choosing which track to record time code onto. They both agree that to prevent frequencies from other recorded material interfering with and corrupting the recorded time code, the adjacent track(s) must be left blank as "guard bands". The first school records on an "outside" track (one at the physical edge of the tape), usually the 24th. This keeps it out of the way and only requires the use of one "guard band". The second school uses the 23rd track to record time code because this offers additional protection against damage from finger prints or spool wear on the edge of the tape. The trade off is that two tracks (the 22nd and 24th) must be wasted as guard bands. All this will be moot soon anyway, when albums are recorded optically!

MIDI Start & Stop

This menu item determines what Music-X will do with MIDI Start, MIDI Stop, and MIDI Continue messages.

What are MIDI Start, MIDI Stop, and MIDI Continue messages?

MIDI Start is a MIDI message that causes all receivers to locate their clocks to the beginning of the song and start playing.

MIDI Stop causes all receivers to halt the advancement of their own clocks.

MIDI Continue can be used after a MIDI Stop, and causes receivers to start playing from their current position without relocating to the beginning of the song.

Three mutually exclusive options are listed in a pop-out menu. The options are described below.

Transmit

While **Transmit** is checked, Music-X will send MIDI Start, MIDI Stop, and MIDI Continue messages whenever appropriate.

MIDI Start is sent when Music-X is put into **PLAY** or **RECORD** modes and the clock is at the beginning of the song.

MIDI Stop is sent whenever Music-X is stopped (i.e. the **STOP** button is clicked, or a STOP event is encountered in one of the sequences).

MIDI Continue is sent when Music-X is put into **PLAY** immediately after being stopped, without having been relocated (i.e. without using the rewind or fast forward arrows, or the **CUE** buttons in the interim).

◆ Note: If Music-X is put into **PLAY** or **RECORD** modes after the clock has been relocated (after rewinding, fast forwarding, or using the **CUE** buttons), and the clock is not at the beginning, a MIDI Song Position Pointer will be sent if enabled.

[see Sequencer Page / Sync Menu / Song Position (below)]

Receive

While this item is checked, Music-X will respond to received MIDI Start, MIDI Stop and MIDI Continue messages as appropriate.

When a MIDI Start message is received, Music-X will locate its clock to the beginning of the song.

When a MIDI Stop message is received, Music-X will stop its clock.

When a MIDI Continue message is received, Music-X will continue playing (using the current **Master Clock** mode) from the current clock position.

Ignore

While this item is checked, Music-X will neither respond to, or send, MIDI Start, MIDI Stop and MIDI Continue messages.

Time Code

This menu item determines which of the four frame rate formats Music-X will use when calculating Absolute time. When synchronizing to external Time Code messages (SMPTE, or MIDI Time Code), the internal Time Code format should be set to match the format used by the external Time Code. When an internal master clock source is used, this setting determines the way the Absolute clock will be updated, and the resolution of Absolute sequences.

Absolute sequences have a resolution of 1/4 frame. The start times of events in Absolute sequences are stored as the number of elapsed quarter frames from the beginning of the piece. Depending on the frame rate used, 1/4 frame is equivalent to these amounts of time:

At 24 FPS, 1/4 frame = 1/96 second.

At 25 FPS, 1/4 frame = 1/100 second.

In Drop Frame and Non-Drop Frame, 1/4 frame = a little over 1/20th second.

◆ Note: Recording an Absolute sequence in one Time Code and playing it back in another would change the timing of the sequence. Be sure to use the same **Time Code** setting when recording that you will ultimately use when syncing to film or video.

This menu item has a pop-out menu listing the four available frame rates.

24 FPS

This is the standard frame rate used with American motion picture film.

25 FPS

This is the standard frame rate used with European television, video, and film.

◆ Note: European Music-X users should set their **Time Code** option to **25 FPS** and resave the Default.Perf so that Music-X boots up configured for PAL compatibility.

Non-Drop Frame

When television was first invented, the standard frame rate was 30 FPS. With the advent of color, it was discovered that the current frame rate of 30 FPS caused interference between the color signals and house current. The solution was to alter the frame rate slightly. The NTSC (National Television Systems

Committee) standardized a new frame rate 29.97 FPS — slightly slower than the original 30 frames per second.

When counting frames with Non-Drop Frame, 30 frames are still called 1 second. But it is actually slightly longer than a second of real time. Due to accumulated errors, at the end of one real hour, 108 frames less than expected have passed. So, Non-Drop Frame format is handy for counting frames but not accurate against the clock over time.

Drop Frame

Drop Frame is the frame rate used for color video in the United States. This will be the most commonly used Time Code for American Music-X users.

Like Non-Drop Frame, Drop Frame format is also rated at 29.97 frames per second. What's different about Drop Frame is that the accumulated errors in frame numbers, due to the slightly slow frequency, are corrected to match real time by dropping some of the frame numbers periodically. At the end of an hour, the total number of frames will still be 108 frames short, but the frame numbers will be correct.

This is the way it works: There is still an orderly progression of frames, one after the other. Each frame has a unique number, and all the frames are numbered in ascending order. However, when numbering each frame, certain frame numbers are skipped. The ones that are left out are the first two frame numbers of every minute except the first minute and every tenth minute after that. That means that a total of 108 frame numbers are left out every hour (just enough to correct the error).

For example, experimenting with Music-X's **Time:** clock, while in Drop Frame, will show that the frame numbers jump from 00:00:59.29 to 00:01:00.02 because there is no frame 00, or 01, at the beginning of minute 01.

Time Code Offset

This option can be used when synchronizing Music-X to incoming SMPTE Time Code or MIDI Time Code messages to shift the beginning of Music-X Time Code against the beginning of external Time Code.

Time Code Offset allows you to start Music-X's clock at any point during the reference Time Code. This can be used, for instance, when recording a short song in the middle of a long film.

Choosing this menu item opens a window in the Sequencer Page showing the current Time Code Offset number in the form of Hours, Minutes, Seconds, and Frames. The Time Code Offset number may be edited using the arrows next to each number.

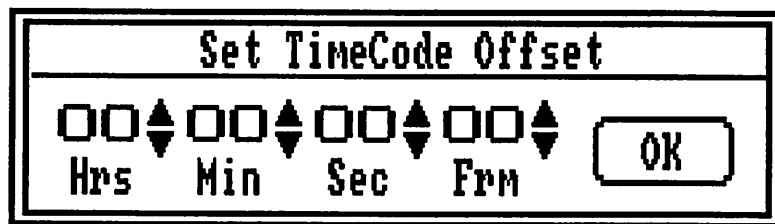


Illustration 6.11: Time Code Offset Window

When done, click **OK** to close the window. The number that you specify will be subtracted from each incoming Time Code message before being used by Music-X.

For example, let's say you're writing some background music to a friend's video. Your friend says the tape is striped with SMPTE and that the music should start 20 minutes into the tape.

- ☐ The video machine must be sending SMPTE messages to the Amiga's SMPTE Reader or to a SMPTE-to-MTC converter connected to the Amiga.

- ☐ Set the Master Clock to **SMPTE Reader**, or **MIDI Time Code** depending on which device you're using.
- ☐ Set the Time Code Offset to 00:20:00.00. Music-X now considers this to be zero (00:00:00.00).
- ☐ Put Music-X in **RECORD**, and run the video from somewhere before the 20 minute mark.
- ☐ As Music-X receives each incoming Time Code message, it subtracts 20 minutes from it and attempts to use it. If a Time Code message appears to be less than zero, it ignores it, and waits for the next one.
- ☐ When Music-X receives the Time Code message specifying 20 minutes, it subtracts 20 minutes from it, sees a message of zero, and says "aha... I can use this." Music-X then begins recording from that point.
- ☐ As you see the action on videotape, play along to match it.
- ☐ When done, **STORE** your music to a sequence and put Music-X in **PLAY**.
- ☐ Rewind the video and play it. Music-X will auto-locate and play back the freshly recorded music along with the video.

MIDI Sync Out

When **MIDI Sync Out** is checked, Music-X will transmit MIDI Clock messages at the current tempo, when in **PLAY** or **RECORD**. This is done regardless of the current **Master Clock** setting.

This is handy when using a drum machine as an external sequencer, that is, with musical drum patterns stored in its own internal memory rather than being triggered by note events from Music-X's own sequences.

For example, to synchronize a separately programmed MIDI drum machine (or other sequencer) from Music-X:

- ☐ Connect a MIDI cable from the Amiga's MIDI OUT to the drum machine's MIDI IN.
- ☐ Set the drum machine to receive MIDI Clocks (most units call this "MIDI sync mode").
- ☐ Set Music-X's **Master Clock** to **Internal**.
- ☐ Turn on **MIDI Sync Out**.
- ☐ Set **MIDI Start & Stop** to **Transmit**.
- ☐ Turn on **Song Position** (see below).
- ☐ Play Music-X from any point in the song. Music-X will send a MIDI Start, or MIDI Song Position Pointer message to locate the drum machine and then begin transmitting MIDI Clock messages at the current tempo.
- ☐ The drum machine will receive the MIDI Clock messages and follow Music-X.

To synchronize a drum machine while Music-X is synchronizing to video:

- ☐ Set up the drum machine as above, and set up the SMPTE Reader to feed Music-X SMPTE Time Code.
- ☐ Set the **Master Clock** to **SMPTE Reader**.
- ☐ Set the **Time Code** type to **Drop Frame** (for color video).

- ☐ Set the **Time Code Offset** if needed to start the song in the middle of the video.
- ☐ Turn on **MIDI Sync Out**.
- ☐ Set **MIDI Start & Stop** to **Transmit**.
- ☐ Turn on **Song Position** (see below).
- ☐ Put Music-X in **PLAY** and start the video from any point.
- ☐ When Music-X receives the first Time Code message, it will auto-locate to the specified point. After auto-locating, Music-X will send a MIDI Start, or MIDI Song Position Pointer message to locate the drum machine and then begin transmitting MIDI Clock messages at the current tempo.
- ☐ As the video is stopped, fast forwarded or rewound, and played again, Music-X will auto-locate to match it, then send Song Position and MIDI Clock messages to locate and sync the drum machine.

Song Position

This Sync menu option enables the sending of MIDI Song Position Pointer messages by Music-X. While this menu option is checked, Music-X will send a MIDI Song Position Pointer message whenever it deems a relocation necessary. This would be whenever **PLAY** is clicked after the current clock setting has been moved (by fast forwarding or rewinding, or by using the **CUE** buttons). The only time a Song Position Pointer would not be sent is when starting from the beginning of a song. In this case, a MIDI Start message would be sent instead (assuming **MIDI Start & Stop** is set to **Transmit**).

◆ Note: Music-X can send Song Position Pointer regardless of the **MIDI Start & Stop** setting.

What is a Song Position Pointer Message?

Song Position Pointer is a MIDI message that describes some point in the song based on the number of sixteenth notes that have elapsed since the beginning. The receiver of a Song Position Pointer message stops whatever it's doing and relocates its clock to the specified point. Once the receiver has located, it can be sent MIDI Clock messages. This process allows the sender and receiver to start together from any point in the song without having to play from the beginning.

◆ Note: The receiving of Song Position Pointer messages is always enabled. Music-X always relocates when it receives a Song Position Pointer message.

Song Position Delay

Song Position Delay is an option that affects the way the **Song Position** option works, and may be used in any situation where Song Position messages are sent. If **Song Position** is not turned on (checked) then the **Song Position Delay** setting is ignored.

The Song Position Delay feature is an allowance made to older drum machines or sequencers that may not have implemented the MIDI specification correctly. It causes Music-X to temporarily delay the sending of MIDI Clocks after a Song Position message has been sent, so that older devices, that ignore their MIDI IN ports while locating, won't miss a few MIDI Clock messages and start out of sync. This feature is only needed when using one of these older units as an external sequencer, that is, with musical drum patterns stored in its own internal memory. If you are triggering the drum machine with note events from Music-X's sequences, then Song Position is not needed.

When a MIDI device receives a Song Position message, it relocates its clock. This is usually done by playing silently, as fast as possible from the beginning of the song. This takes time. The further into the song it must go, the longer it takes to locate.

Here's a scenario of the problem:

- ☐ After fast forwarding or using a **CUE** point, **PLAY** is clicked.
- ☐ Music-X notices the new clock time and begins locating silently to the new position, playing through any Time Signature Changes and calculating the number of 16th notes that have elapsed.
- ☐ Music-X finishes auto-locating and sends a Song Position Pointer message specifying the accumulated 16th notes.
- ☐ The drum machine receives the Song Position Pointer message and begins to auto locate.
- ☐ Music-X begins playing and sending MIDI Clock messages.
- ☐ The drum machine is still locating and misses the first few MIDI Clocks. (MIDI specs say that it should keep track of incoming MIDI Clocks while locating.)
- ☐ The drum machine finishes auto locating and begins playing from the previously specified point at the rate dictated by the incoming MIDI Clock messages.
- ☐ Music-X has meanwhile played ahead.
- ☐ The two units remain perfectly out of sync until the end of the song!

One solution would be to click **PAUSE**, and then **PLAY**, and wait for the receivers to catch up before turning **PAUSE** off to play again. **Song Position Delay** essentially does this automatically.

Here's a scenario of the solution using **Song Position Delay**:

- ☐ After fast forwarding or using a **CUE** button, **PLAY** is clicked.
- ☐ Music-X notices the new clock time and begins locating silently to the new position; playing through any Time Signature Changes and calculating the number of 16th notes that have elapsed.
- ☐ Music-X finishes auto-locating and sends a Song Position Pointer message specifying the accumulated 16th notes.
- ☐ Music-X begins waiting for a period of time based on the magnitude of the Song Position message.
- ☐ The drum machine receives the Song Position Pointer message and begins to auto locate.
- ☐ The drum machine finishes auto locating and waits for MIDI Clock messages to advance its clock.
- ☐ Music-X begins playing and sending MIDI Clock messages.
- ☐ The drum machine begins playing at the rate dictated by the incoming MIDI Clock messages.
- ☐ The two units remain in sync until the end of the song.

The process described above is what happens when using **Song Position Delay** while the **Master Clock** setting is set to either **Internal** or **Video Clock**. While using these **Master Clock** sources, Music-X can wait as long as it sees fit before playing, because it's in control.

Song Position Delay is implemented slightly differently when using either **MIDI Time Code**, or **SMPTE Reader** as the **Master Clock**. When using either of these **Master Clock** settings, it's impossible to tell what the first incoming message will be. This is because the timing source can relocate independently of Music-X. Music-X must follow it. When Music-X receives the first incoming Time Code message, it converts it internally to a song location. It then sends a Song Position Pointer message specifying one measure later than the current position, and starts playing. When one measure has passed, Music-X starts sending MIDI Clocks (assuming that **MIDI Sync Out** is switched on).

Here's a scenario of the behavior of Song Position Delay when synchronizing to a Time Code.

- ☐ **PLAY** is clicked.
- ☐ The tape recorder or other device is started.
- ☐ Music-X begins receiving Time Code messages and figures out where in musical time it should be, based on the Absolute time specified in the Time Code message.
- ☐ Music-X begins silently locating to the new position; playing through any Time Signature Changes and calculating the number of 16th notes that have elapsed.
- ☐ Music-X finishes locating and sends a Song Position Pointer message specifying one measure greater than the current measure.

- ☐ Music-X continues playing but doesn't transmit MIDI Clocks.
- ☐ The drum machine finishes auto locating and waits for MIDI Clock messages to advance its clock.
- ☐ After Music-X reaches the next measure, it begins transmitting MIDI Clock messages.
- ☐ The drum machine begins playing at the rate dictated by the now incoming MIDI Clock messages.
- ☐ The two units remain in sync until the end of the song.

Modules Menu

A Modules menu is found on every page in Music-X (except the Keymap Editor Page and the Protocol Editor Page, which are modules themselves). A module is a small program loaded into Music-X to execute a particular function. Six modules come with Music-X as released. More will become available in the future. Following are descriptions of the two modules that appear in the Sequencer Page / Modules Menu. Other modules are described elsewhere in the manual.

NewCLI

This module calls the standard Amiga program "NewCLI" which opens a text window titled "New CLI" in the Workbench screen.

CLI stands for "Command Line Interpreter". The CLI window is used to give written commands to the AmigaDOS operating system. This gives you another way to communicate with the Amiga to do things like look at the contents of disks, run programs, delete files, etc.

Move between the Workbench and Music-X screens as needed using the front-to-back gadgets found on most screens and windows.

◆ Note: Learn how to use the CLI by reading an AmigaDOS Manual.

NewShell

This module calls the Amiga program “NewShell” which opens a text window titled “AmigaShell” in the Workbench screen. A Shell is basically an improved CLI. Move between the Workbench and Music-X screens using the front-to-back gadgets.

◆ Note: Learn how to use the NewShell by reading an AmigaDOS Manual.

Transport Window

In the top left corner of the Sequencer Page is the Transport window containing a cluster of controls modeled after the buttons on an analog tape machine. These Transport Controls are activated by clicking on them, and are used to start the clock, stop the clock, and relocate the clock to different portions of the song. Following are descriptions of those buttons.

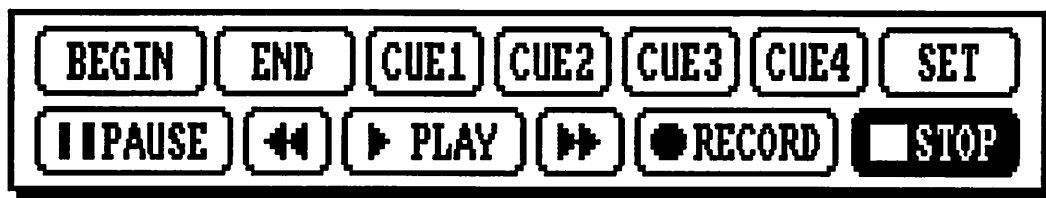


Illustration 6.12: Transport Window

BEGIN

("B" from the keyboard)

This button relocates the clock to the beginning of the song. This button can be used while the clock is stopped or while the clock is running. If used while in **RECORD** mode, the Record Buffer is erased, the clock is moved to the beginning of the song, the sequencer backs up to play the Count-In measures (see below), and recording starts again.

END

("E" from the keyboard)

This button relocates the clock to the last previously recorded point in the song. The **END** button is self-adjusting, and automatically recalculates its position each time a longer sequence is recorded. However, it can also be set manually using the **SET** button (see below).

This button can be used while the clock is stopped or while the clock is running. If used while in **RECORD** mode, the Record Buffer is erased, the clock is moved to the end of the song, the sequencer backs up to play the Count-In measures (see below), and recording starts again.

CUE1 CUE2 CUE3 CUE4

("1", "2", "3", and "4" from the keyboard)

These user-definable **CUE** buttons are designed to emulate the auto-locate functions of a professional multi-track tape recorder. They can be used instead of Fast Forward or Rewind to quickly relocate the clock to preset points. Those points can be set to the start of song sections or to any point in the song using the **SET** button (see below).



("P" from the keyboard)

(ESC (Escape key) = All notes off)

(SHIFT-ESC = Super All notes off)

This button starts the sequencer from the current clock position. If the clock is not at the beginning, then Music-X auto-locates by playing from the beginning, as fast as possible, until it gets to where the clock is, before playing normally. During auto-locate, Note events are silenced, but all other events are played normally (albeit quickly). This is so that things like Program Changes, Pitch Bend, Control Changes, and other events that might affect the sound will be correctly updated to their most recent values when the clock begins playing normally.

If **PLAY** is clicked while in **RECORD** mode, recording is stopped and the clock continues running.

If at anytime during play, a "MIDI stuck-on note" occurs, remember that pressing the Amiga's escape key (at the upper left corner of the keyboard) sends a MIDI All Notes Off message. Holding the SHIFT key, while pressing the escape key initiates a "Super All Notes Off", which actually sends a MIDI Note-Off message for each of the 128 possible notes on all 16 MIDI channels. This feature is available from all Pages in Music-X.



("]") (right bracket) from the keyboard)

The Fast Forward Button is marked with two right-pointing arrows. Depending on which clock is currently chosen, clicking this button will advance the clock in increments of 1 measure, or 1 minute. Holding the button moves the clock continuously.

The Fast Forward button cannot be used during **PLAY** or **RECORD** modes.

◆ Note: The clocks can also be changed in smaller increments by using the green arrows in the clock window.

[see Sequencer Page / Clock Window]



("R" from the keyboard)

The basic process in MIDI recording is to capture the incoming data stream from the mother keyboard (or whatever device is hooked up to the computer's MIDI IN port), time-stamping each arriving event with the current clock time.

In Music-X, incoming data is first passed through the Filters and Keymaps Pages, and is then held in the temporary Record Buffer (also known as the Edit Buffer). When you finish recording (when **STOP** is clicked), the Edit Buffer is closed and awaits a **STORE** command. Clicking **STORE** moves the newly captured (recorded) data into the current sequence freeing the Record Buffer to record again.

[see Filters Page chapter]

[see Keymap Editor chapter]

[see Sequencer Page / STORE]

RECORD, when clicked, calls the **RECORD SEQUENCE** window offering this list of record parameters:

Absolute Time Seq
Punch-In Manual
Punch-In Auto
Loop Mode ON
Mix Down Mode

The **CUE** buttons can be used while the clock is stopped or while the clock is running. If used while in **RECORD** mode, the Record Buffer is erased, the clock is moved to the CUE point, the sequencer backs up to play the Count-In measures (see below), and recording starts again.

The default setting for the **CUE** buttons is the beginning of the song.

CUE button locations can be saved with performance files.

[see Sequencer Page / File Menu / Save Performance]



SET

("=" (equals sign) from the keyboard)

This button is used to manually set the locations for the **CUE** buttons (and the **END** button if you wish). **SET** can be used while the clock is running or while the clock is stopped.

Here's how to set a **CUE** point using the **SET** button.

- ☐ Click **SET**. **SET** will appear highlighted; it is now "readied".
- ☐ Click on one of the **CUE** buttons (or the **END** button). The **CUE** button will be set to the current clock position. **SET** will return to normal appearance.

You can change the cue points at any time by repeating the above steps.



II PAUSE

(SPACE BAR from the keyboard)

As in a conventional tape recorder, you can temporarily halt the playing or recording of a song with the **PAUSE** button.

Click **PAUSE**. **PAUSE** will become highlighted, and the clock will temporarily stop. You can resume playing by clicking **PAUSE** again. When **PAUSE** is released, the sequencer does not need to relocate as it would with the series of commands **STOP** and **PLAY**.

If you click **PAUSE** while recording, the Record Buffer is not erased. Releasing **PAUSE** will continue recording from that point.

PAUSE does not automatically cut off any currently sounding notes; it holds the clock in its current state.



("[" (left bracket) from the keyboard)

The Rewind Button is marked with two left-pointing arrows. Depending on which clock is currently chosen, clicking this button will move the clock back in increments of 1 measure, or 1 minute. Holding the button moves the clock continuously. If you are already at the beginning, then nothing will happen!

The Rewind Button cannot be used during **PLAY** or **RECORD** modes.

◆ Note: The clocks can also be changed in smaller increments by using the green arrows in the clock window.

[see Sequencer Page / Clock Window]

Mute Target Seq.
Punch Overlay
Punch-In:
Punch-Out:
Count-In Bars
Record Bars

Descriptions of these parameters can be found in the next manual subchapter.

[see Sequencer Page / RECORD SEQUENCE window]

If you change your mind and decide not to record anything yet, click **CANCEL** to safely close the window and return to the Sequencer Page. Otherwise, after the parameters are set, clicking the **GO** button, or playing a note on the mother keyboard, starts the clock and begins recording.



("S" from the keyboard)

The **STOP** button does what you'd expect; the clock is halted as in a normal tape player. If you were previously in record mode, the Record Buffer is closed and now awaits a **STORE** command.

[see Sequencer Page / STORE]

STOP also acts like an intelligent **CUE** point in that it remembers the clock position at the last time it was clicked. Test this by clicking **PLAY**, then **STOP**. Note the clock time. Click **BEGIN** to relocate the clock to the beginning. Click **STOP** again. Note that the clock has relocated, reset to where you last left off.

Any currently sounding notes are automatically cut off when **STOP** is clicked.

RECORD SEQUENCE Window

The **RECORD SEQUENCE** window appears when the **RECORD** button is clicked. Music-X offers a plethora of recording modes and options. This window is used to set these options before recording.

RECORD SEQUENCE

- ☐ Absolute Time Seq
- ☐ Punch-In Manual
- ☒ Punch-In Auto
- ☐ Loop Mode ON
- ☐ Mix Down Mode
- ☐ Mute Target Seq.
- ☒ Punch Overlay

Punch In: 0004.01.000
Punch Out: 0008.01.000

Count-In Bars: 02 ▲▼
Record Bars: 4095 ▲▼

Press GO or play
any note to start.

GO **CANCEL**

Illustration 6.13: Record Sequence Window

In the window is a column of option switches that are turned on and off by clicking on them. When an option is turned on, its button appears highlighted in light blue, and it remains active until turned off. When an option is turned off, its button

appears dark blue and hollow. By default, they are all turned off until set by you. More than one option may be turned on at the same time, although certain combinations are not allowed, and turning on some buttons will turn off others. Following are descriptions of the controls in the **RECORD SEQUENCE** window.

Absolute Time Seq

Read this option as “Absolute Time Sequence”.

When this switch is turned on (appearing highlighted), the new sequence will be recorded in Absolute Time format. That is, the start times of events will be stored in Hours, Minutes, Seconds, and Quarter Frames.

Absolute sequences are handy for causing things to happen at particular moments, as in doing sound effects for film. An absolute sequence runs at the same rate regardless of the Tempo setting, because it’s based on real clock time, instead of metered musical time.

If this switch is turned off (appearing hollow), then the sequence will be recorded in Relative Time format. That is, the start times of events will be stored in Measures, Beats, and Clocks according to the time signature.

Punch-In Manual

Turning this switch on puts Music-X in Manual Punch-In mode. Punch-In is an analog tape recording technique where a short segment of a previously recorded performance is rerecorded, erasing and replacing a mistake with a correction. Manual Punch-In Mode lets you use the RETURN key to manually punch into a prerecorded sequence. The sequence to be punched into is called the “target sequence”. Recording in Punch-In mode creates a new sequence in the Record Buffer that contains some data from the target sequence and new data from the mother keyboard.

Example:

- ☐ Choose the target sequence by making it the current sequence (highlighting it in the sequence list).
- ☐ Click **RECORD**.
- ☐ When the **RECORD SEQUENCE** window appears, turn on the **Punch-In Manual** switch.
- ☐ Click **GO** to close the **RECORD SEQUENCE** window and start the clock.
- ☐ Data from the target sequence is copied into the Record Buffer as it plays.
- ☐ Press the RETURN key to punch in (begin recording new data), and play some new notes on the mother keyboard. Data from the mother keyboard is now being recorded into the Record Buffer; data from the target sequence is not.
- ☐ Press the RETURN key again to punch out (stop recording new data). Data from the target sequence is again being copied into the Record Buffer; data from the mother keyboard is not.
- ☐ Repeat the last two steps as often as needed.
- ☐ Click **STOP**. The rest of the target sequence is automatically copied into the Record Buffer.
- ☐ The Record Buffer now contains the new sequence that is a combination of the target sequence and the new data.
- ☐ **STORE** the Record Buffer back into the target sequence to replace it, or **STORE** it elsewhere and keep the original intact.

 Punch-In Auto

While this switch is turned on (appearing highlighted), Music-X is in Auto Punch-In mode. This is similar to Manual Punch-In except that the Punch-In points may be preset and executed automatically.

◆ Note: Punch-In points are set from the Sequencer Page / Options Menu, or in real time with the F1 and F2 keys on the Amiga keyboard.

[see Sequencer Page / Options Menu / Set Punch-In]

Turning this switch on also turns on the **Punch-In:** and **Punch-Out:** indicators (see below) that show the clock positions that delimit the punch period.

This feature can be handy when punching into an especially tight time space where a mistake could erase needed data, or when you need to keep your hands free to play.

◆ Note: It's still possible to punch in manually, using the RETURN key (or from the mother keyboard using a keymap) while in Punch-In Auto mode.

Remember that any recording, including recording using the Punch-In modes, puts new information into the Record Buffer; it does not alter the original sequence. You can still use the **PREVIEW** button to try out the new recorded version of a sequence before storing it over the original. You may also store the new version in a different sequence and keep the original as a backup.

 Loop Mode ON

While this switch is on, Music-X will be in Loop Record Mode. This is used in conjunction with the **Record Bars** parameter (see below). What this does is to record normally from the beginning of the piece until the number of **Record Bars** is reached. At that time the clock is automatically reset to the beginning of the song,

the sequencer backs up to play the **Count-In Bars** (see below) and recording continues. The Record Buffer is not erased, and new data is continuously recorded.

This feature is handy for recording many passes over the song without having to stop to store each pass. Often this approach can capture a better musical performance by not breaking the flow for the performer.

Once a good musical performance has been captured, the sequence can be stored and edited to remove the unneeded passes. To aid in editing a sequence recorded in Loop Mode, **SPLICE** events are added to the sequence as markers at the beginning of each repeated section.

[see Event Editor / Event Types / SPLICE]

Mix Down Mode

When this switch is turned on (appearing highlighted), data being played by other sequences will be recorded into the Record Buffer. Input from the mother keyboard is ignored. This can be used to mix the data from several looping sequences into one continuous sequence, or to overdub multiple instruments onto a “composite” sequence (one containing data for more than one instrument). This is analogous to mixing down many tracks of a multi-track tape recorder to a single track. This is handy for doing edits that affect an entire song. For example, you could mix down a repeating bass line into one continuous sequence and then scale its velocity to effect a fade out.

◆ Note: Data from Absolute sequences cannot be mixed down into Relative sequences, and vice versa. When recording an Absolute sequence in Mix Down Mode, data from Relative sequences will be heard but not recorded. When recording a Relative sequence in Mix Down Mode, data from Absolute sequences will be heard but not recorded.

◆ Note: The outputs of sequences being played during Mix Down Mode are first passed through the Channelizer. Data recorded during Mix Down Mode will have the channel assignments as changed by the Channelizer.

[see Sequencer Page / Options Menu / Output Channels]

Mute Target Seq.

Read this as “Mute Target Sequence”.

The target sequence is the sequence to be rerecorded or punched into (chosen as the current sequence before recording). If a blank sequence is the target then the position of this switch is ignored.

If this switch is turned on before recording, then the target sequence will be muted during recording. If used during a punch-in then the target sequence is only muted during the actual punch.

If this switch is left turned off, then the old sequence can be heard while recording. Sometimes it can be handy to leave the old sequence on as a timing reference, or to hear the mistake as you correct it. If left off in Mix Down mode then it can be used to overdub onto a composite sequence (one containing data for more than one instrument).

Punch Overlay

This switch is used in conjunction with the Punch-In modes. It defeats the erase function when punching.

If **Punch Overlay** is turned on, then data from the original sequence is not erased during a punch-in. New data from the mother keyboard is added to the target sequence rather than replacing the data that was there.

This feature can be used to build a sequence in sections by overlaying smaller segments until the part is complete.

Punch In: 0004.01.000

This indicator shows the current Punch-In point when using **Punch-In Auto**. If not using **Punch-In Auto** then no numbers are shown.

This is an indicator only, and cannot be set from within the **RECORD SEQUENCE** window.

◆ Note: The Punch-In point is set from the Sequencer Page / Options Menu, or in real time with the F1 key on the Amiga keyboard.

[see Sequencer Page / Options Menu / Set Punch-In]

Punch Out: 0008.01.000

This indicator shows the current Punch-Out point when using **Punch-In Auto**. If not using **Punch-In Auto** then no numbers are shown.

This is an indicator only, and cannot be set from within the **RECORD SEQUENCE** window.

◆ Note: The Punch-Out point is set from the Sequencer Page / Options Menu, or in real time with the F2 key on the Amiga keyboard.

[see Sequencer Page / Options Menu / Set Punch-Out]

Count-In Bars: 02 ◆

With this parameter, you can set up to 12 measures of Count-In time. This causes the sequencer to back up and play a number of measures before the normal starting

time. This lets you hear a little music, to “get the feel” before actually recording. Any events recorded during the Count-In measures are placed on the down beat of the first real measure. This means, if you play the first chord a little early, you still get all the notes.

- Click and hold on the arrows next to the indicator to increase or decrease the current number of Count-In Bars. Or click on the indicator itself to activate the cursor, then edit it using the Amiga’s numeric keys.

Example:

- ☐ From the Sequencer Page, set the clock to the beginning of measure 9.
- ☐ Click **RECORD** and use the record requester to set the **Count-In Bars** to 4.
- ☐ Click **GO** to close the requester. The sequencer will auto-locate to 4 measures before the beginning of measure 9 (to the beginning of measure 5) and begin playing.
- ☐ The sequencer will play through the 5th, 6th, 7th, and 8th measures as Count-In Bars.
- ☐ When it reaches the beginning of measure 9, it will begin recording normally.

If you are recording from the beginning of the piece, you can still use the Count-In Bars parameter. The sequencer will back up into negative time, and play from there. Sequences whose measure offsets have been set to a negative value can be heard during the Count-In. (The typical procedure when creating a metronome sequence is to set it to repeat indefinitely and set its measure offset to a negative number.)

[see Bar Editor / Event Types / Set Repeats]

[see Sequencer Page / Sequence List / Offset Arrows]

◆ Note: When Music-X is synchronizing to an Absolute-Time source (one inherently containing real time location information i.e., SMPTE Reader, MIDI Time Code, or Video Clock), it must “chase” that source and cannot arbitrarily back up the clock. Count-In Bars are therefore not available when using any of these master clock settings. In this case, you can use **Punch-In Manual** or **Punch-In Auto** settings and “pre-roll” your time source to give yourself some count-in time.

[see Sequencer Page / Sync Menu / Master Clock]

Record Bars: 4095 ◆

This parameter is used to set the number of measures the sequencer will record before automatically stopping the clock. This can be handy when recording sections of a set length. If you’re improvising, or are not sure how long the section will be, just leave the setting at maximum and stop the sequencer manually.

Click and hold on the arrows next to this indicator to change the current setting. Or click on the indicator itself to activate the cursor, and edit the number using the Amiga’s numeric keys.

The default setting for **Record Bars** is the maximum length allowed, 4095 measures. Using a typical time signature and tempo, this equates to over two hours of music. Enough to score the average movie.

After setting all of the above mentioned parameters in the **RECORD SEQUENCE** window, start the clock by clicking **GO**, or pressing the space bar, or pressing a key on the mother keyboard (it won’t be recorded).

Music-X will then start recording from wherever the clock is (offset by the **Count-In Bars** parameter). If the clock is not at the beginning, then Music-X silently locates to the current clock setting before beginning to record.

Clock Window

The Clock window is found directly below the Transport window. It contains two digital clock faces.



Illustration 6.14 Clock Window

The first, titled **Clock:**, is the Relative Time clock and reads out in Measures, Beats, and Clocks. Clocks are the smallest measurement of Relative (musical) time possible in Music-X, and are equal to 1/192 quarter note.

The second, titled **Time:**, is the Absolute Time clock and reads out in Hours, Minutes, Seconds, and Frames. The number of frames per second depends on the Time Code being used. Frames are separated by a period in Drop Frame mode, and by a colon in 24, 25 and Non Drop Frame modes.

When the sequencer is in **PLAY** or **RECORD** modes, these clocks change in real time to show the current musical position and the current elapsed time. While the sequencer is stopped, either of these clocks may be changed manually to relocate the sequencer.

[see Sequencer Page / Sync Menu / Time Code]

To relocate the Relative clock:

- ☐ Change the Relative clock in increments of 1 measure by first clicking on the word **Clock:** and then using the Fast Forward or Rewind buttons in the Transport window.

[see Sequencer Page / Transport Window / Rewind Button]

[see Sequencer Page / Transport Window / Fast Forward Button]

- ☐ Change the Relative clock in increments of 1 beat by clicking and holding the green arrows directly to the right of the clock face.
- ☐ Change the Relative clock in increments of 1 clock by clicking and holding the green arrows to the far right of the clock face.

To relocate the Absolute clock:

- ☐ Change the Absolute clock in increments of 1 minute by first clicking on the word **Time:** and then using the Fast Forward or Rewind buttons in the Transport window.

[see Sequencer Page / Transport Window / Rewind Button]

[see Sequencer Page / Transport Window / Fast Forward Button]

- ☐ Change the Absolute clock in increments of 1 second by clicking and holding the green arrows directly to the right of the clock face.
- ☐ Change the Absolute clock in increments of 1 frame by clicking and holding the green arrows to the far right of the clock face. (A frame is a subdivision of a second used in the film industry, relating to the rate at which frames of film are displayed.)

The length of a piece can found by fast forwarding to the end of the song, clicking **PAUSE**, clicking **PLAY**, and reading the **Time:** clock.

Tempo Window

Directly below the Clock window is the Tempo window. It has three features (one is hidden). Following are descriptions of those features.

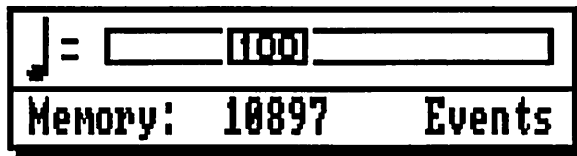


Illustration 6.15 Tempo Window

Tempo Slider

This slider-indicator shows the current tempo setting in quarter notes per minute (rather than “beats” per minute which would change with different Time Signatures). Available tempos range from 10 to 300. The slider can be manually adjusted at any time, including while in **PLAY** and **RECORD** modes.

If Tempo Change events are imbedded in the sequences, then the Tempo slider will move in real time as they are played. If no Tempo Change events are imbedded in the sequences, then the slider setting determines the tempo for the whole tune.

[see Bar Editor / Event Types / Tempo Change]

Drag the slider to make coarse adjustments. For finer adjustments, click on either side of the slider to increase or decrease the tempo in increments of 1 beat per minute. Holding the mouse button, after clicking on either side of the slider, will cause the tempo to increase or decrease continuously.

Memory:

Also found in the Tempo window, below the Tempo slider, is the **Memory:** indicator.

When first booted, Music-X grabs as much available memory as it can, to reserve space for musical events. This space is called the “buffer”. The **Memory:** indicator shows the approximate number of events that can still be recorded into the buffer before running out of room. This number is calculated by dividing the amount of free bytes by ten. It takes ten bytes of memory to store a Note-On and its associated Note-Off along with their timing bytes.

The size of this number will depend on the amount of RAM in your system as well as whether or not you are running other programs in your Amiga at the same time. With a full 8 meg Amiga system, there is room for approximately 800,000 notes.

♦ Note: If you want to choke down the Buffer size in Music-X (the maximum amount of RAM that Music-X grabs when it boots up) you can do so by setting one of the tool types in the info file of Music-X or its Performance files to “BUFFER=10000” or whatever size you want (in bytes). You may also specify the number using an optional “k” to represent kilobytes, as in “BUFFER=10k” (meaning ten kilobytes).

[see Advanced Users / Workbench Tool Types]

Error Indicator

There is a hidden error indicator to the right of the **Memory:** number field. If you ever see the word **ERR:** and a number, this means some data was skipped when reading or writing to the serial port. This is similar to the “MIDI Error” that some keyboards report when overloaded.

Time Signature Window

The Time Signature window is found to the right of the Tempo window and below the Clock window. It contains an indicator showing the current time signature. The indicator may be changed manually, using the green arrows found at the right side of the gadget.



Illustration 6.16 Time Signature Window

If Time Signature Change events are found in a sequence during **PLAY** mode, then this indicator will change in real time as the events are played. If there are no Time Signature Change events in the sequences then this setting determines the global Time Signature for the whole piece.

As in normal music theory, the top number in the time signature shows the number of beats in a measure and the bottom number shows which note value represents one beat.

There are four green arrows in the window: one to increase, and one to decrease each of the two numbers in the time signature. The possible bottom numbers are: whole notes (1), half notes (2), quarter notes (4), eighth notes (8), and sixteenth notes (16). The top number can be any number of beats per measure from 1 to 64, but cannot be greater than four times the bottom number. Possible time signature therefore range from 1/1 to 64/16.

[see Bar Editor / Event Types / Time Signature Change]

Sequence List Window

The bottom half of the Sequencer Page is used by the Sequence List window. The sequence list holds entries for the 250 available sequences in Music-X.

⬆	Seq	Men	Bars	Channels	CST	Time	Sequence Name	Out	<< >>
	001	112	1	1	C--	Rel	Metronome Sequence	INT	-0008
	002	-	--	-	-	-	*		---
	003	-	--	-	-	-	*		---
	004	-	--	-	-	-	*		---
	005	-	--	-	-	-	*		---
	006	-	--	-	-	-	*		---
	007	-	--	-	-	-	*		---
	008	-	--	-	-	-	*		---
⬇	009	-	--	-	-	-	*		---

Illustration 6.17: Sequence List Window

Scroll through the list with the CursorUp and CursorDown keys or by dragging the scroll bar to the left of the list. Using the CursorUp and CursorDown keys while holding the SHIFT key causes the Sequencer to scroll through the sequence list by pages. One sequence will always be highlighted; this is the “current sequence”. Any operations called from the Sequencer Page or its menus that affect a sequence (such as **RECORD**, **EDIT**, **DELETE**, **STORE**, **Copy Sequence**, etc.) will affect the current sequence.

The Sequence List has a number of fields that describe things about the sequences. The names of the fields are written along the top of the list. Following are descriptions of those fields.

Seq

Read this as “Sequence Number”.

This left-most field shows the sequence number for each sequence. Available

sequences are numbered from 1 to 250. These numbers are used by the software to identify the sequence and cannot be changed.

Mem

Read this as “Memory Used”.

This field shows the exact amount of RAM used by each sequence, in bytes (as opposed to events).

Bars

This field shows the length of each sequence in measures, or “bars”. This is determined by the location of the last event found in each sequence. (That means, that if there are events after the END line, this number will show the total length of the sequence rather than the point at which the END line occurs.)

Channels

This field is used to show which MIDI channels are used by each sequence. Composite sequences have more than one MIDI channel. Single channel numbers appear separated by commas; a dash between two numbers means all the channels between those two numbers are used. Example: “1,3-5” means channels 1, 3, 4 and 5 are used.

CST

Read this as “Control, Sequence, and Time.”

This field is used to indicate the presence of Music-X events in a sequence. Music-X events are those that control the sequencer internally. Depending on the events present, the characters: **C**, **S**, or **T** are displayed. You can quickly see which sequences contain tempo changes (**T**) or are triggering other sequences (**S**) or contain other control events (**C**). Here’s the breakdown:

T = (T)empo Change events or (T)ime Signature events. Both of these events affect the timing of the sequencer. Think “T” for time.

S = (S)equences Play event. This would indicate a “control” sequence (one that triggers other sequences). Think “S” for sequence.

C = (C)ontrol events. All the other Music-X control events are grouped together under **C**. The most common will probably be Set Repeats (REPT). The other events that cause **C** to appear are: Mute Track (MTRK), Solo Track (STRK), Mute Sequence (MSEQ), Solo Sequence (SSEQ), and STOP. Think “C” for control.

Time

This field indicates which time format the sequence was recorded in. It reads **Rel** for Relative Time format, or **Abs** for Absolute Time format.

The choice to record a sequence in either format is made in the **RECORD SEQUENCE** window before recording.

[see Sequencer Page / RECORD SEQUENCE Window / Absolute Time Seq]

Sequence Name

This field provides a 27-character space to name each sequence. Do be creative. Example: “Best sounding Bass run yet!”

Click on the name at any time (including during **PLAY** or **RECORD** modes) to activate the red text-editing cursor. Use the DEL (delete) key to erase the old name. Edit the new name using the Amiga keyboard. Press RETURN when done (or simply choose any other function).

Empty sequences cannot be named.

Out

Read this as “Output Bank”.

This field indicates the bank to which the data in each sequence will be directed when played. Two banks are available: the bank of 16 external MIDI channels, and the bank of 16 internal Amiga Samples. Each sequence is directed either externally or internally, not both. However, both types of sequences may be played at the same time during a performance.

Change the setting of this field by clicking on it. It will alternate between **INT** and **Ex1**.

When the **Out** field is set to **INT**, the sequence will be played using the internal Amiga Samples. Notes on channel 1 will be played using the first Amiga Sample in memory. Notes on channel 2 will be played using the Second Amiga sample in memory, and so forth.

[see Amiga Samples chapter]

When the **Out** field is set to **Ex1**, the sequence will be played on the MIDI bus. **Ex1** means “External Bank 1”. Right now there is only one MIDI bus with 16 channels. However, this parameter is being left open for future expansion if hardware is developed to allow the Amiga to work with multiple MIDI buses. In that case the choices might include **Ex2**, **Ex3**, **Ex4**, etc.

◆ Note: If for some reason you want a sequence to be played internally and externally at the same time, make a copy of the sequence, then set one copy to **INT** and one copy to **Ex1**.

[see Sequencer Page / Options Menu / Copy Sequence]

Offset arrows

This field is used to do two things. First, to turn a sequence off so that it will not play unless triggered by another sequence using a Play Sequence (PSEQ) event. Second, if the sequence is not turned off, to set the sequence's measure offset to cause the sequence to delay or anticipate entry by a number of measures.

[see Bar Editor / Event Types / Play Sequence]

This field is edited in two ways as well. First, clicking on the field itself turns the sequence off and on. The field will read **Off** when the sequence is off. It will contain a number when the sequence is on. Second, while the sequence is on, clicking and holding on either set of offset arrows directly above the sequence list will increase or decrease the offset number.

When you first record and **STORE** a sequence, this field is set to **0000**. What this means is that the sequence is turned on and will auto-start and play back exactly as recorded the next time **PLAY** is clicked. Increasing the offset number will cause the sequence to wait that same number of measures before starting. Decreasing the offset to a negative number will cause the sequence to start playing before zero and be heard during the Count-In Bars when recording. (Negative settings are normally used only for metronome sequences. If the clock is started at the origin, then sequences set to start before zero will seem to start somewhere in the middle of their data.)

♦ Note: Turning off this field during **PLAY** will not alter the music. What it actually does is set up a flag that tells the sequence not to auto-start (play itself) the next time **PLAY** is clicked. To temporarily mute a sequence during **PLAY**, click on its name where it appears in the track window in the upper right corner of the Sequencer Page.

[see Sequencer Page / Track Window]

Track Window

The Track window is found in the upper right corner of the Sequencer Page. This window is used to show the status of each of the twenty tracks in Music-X.

What are tracks?

A track is something that plays a sequence. Think of a sequence as being like a piece of tape and a track being like the playback head. Only one piece of tape may be played by a playback head at a time, but many different pieces of tape may pass by it during its lifetime; a track may only play one sequence at a time, but is not dedicated to that one sequence. Consider also that a drum machine can be thought of as a one-track sequencer that plays many separate short sequences one after the other to create a longer song. Music-X is a twenty-track sequencer. Any twenty sequences may be playing at any one time. As a sequence is triggered, the next available track begins to play it and its name appears in the track window. When a sequence stops, its name disappears from the track window, and the track that was playing it becomes free to play other sequences as needed.

♦ Note: Users of other sequencers may note that this is a slight departure from the popular terminology, where a track and a sequence are the same thing.

The track window has two fields titled **Bar**, and **Seq**. The **Seq** field shows the name of the sequence currently being played by each track. The **Bar** field shows which measure of each sequence is currently being played. A sequence that has been offset to start late will show a negative **Bar** number while it counts down before it enters. As a sequence repeats, its bar count will reset. When one of the tracks is playing an Absolute Sequence, its Bar field reads in total seconds of elapsed time.

[see Sequencer Page / Sequence List / Offset Arrows]

Temporary live muting and unmuting of tracks during **PLAY** or **RECORD** is done by clicking on them in the track window. They turn green and shut up!

◆ Note: Tracks and sequences can also be muted by imbedding Mute Sequence (MSEQ) and Mute Track (MTRK) events in a sequence and playing that sequence. If you set up a keymap to do so, you can trigger live muting from the mother keyboard and even record your “dub mix” in its own sequence.

[see Keymap Editor chapter]

A track playing a sequence that's being punched will temporarily turn red to indicate that it's being punched.

During **PLAY**, a track playing a sequence that would be punched will briefly turn blue to indicate when the punch would have occurred had it been in **Record** mode.

A track that is playing no sequence will appear blank.

STORE

(SHIFT-“S” from the keyboard)

Just about in the middle of the Sequencer Page is the **STORE** button. This button is used to move the contents of the Edit Buffer into sequence memory. The Edit Buffer may contain a freshly recorded musical performance or a previously edited sequence. If the current sequence is not empty when the **STORE** button is clicked, then a safety prompt will appear to prevent accidental erasure (pictured below).

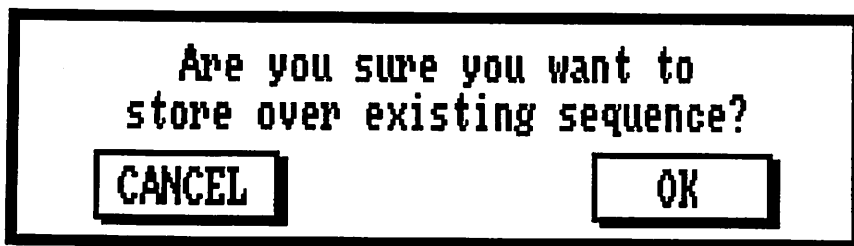


Illustration 6.18 Store Safety Prompt

Clicking **OK** will overwrite the current sequence with the data in the Edit Buffer. Any data previously in the current sequence will be lost. Once the Edit Buffer is stored, it is also emptied.

◆ **Note:** The Edit Buffer is also used by other pages in Music-X. A safety prompt may sometimes appear in other pages when a requested operation would lose unsaved data still in the buffer. Example: Record a sequence into the Edit Buffer but don't **STORE** it. Go to the Librarian Page and attempt to open a new Library Page. A prompt will appear to warn you about the data still in the Edit Buffer.

EDIT

(SHIFT-"E" from the keyboard)

Directly below the **STORE** button is the **EDIT** button. The **EDIT** button is used to edit the current sequence.

When you click **EDIT**, a copy of the current sequence is written into the Edit Buffer, and you are moved into the Bar Editor or the Event Editor (whichever was last used; the default is the former) where you can manipulate the Edit Buffer.

**WARNING: The Edit Buffer currently
contains unsaved data.**

CANCEL

DISCARD

STORE

EDIT IT

Illustration 6.19 Edit Safety Prompt

If the Edit Buffer is not empty when you click **EDIT**, a safety prompt appears to prevent its accidental erasure and offer choices of what to do with the data within. The choices in the safety prompt are described below.

CANCEL

If you choose **CANCEL** you are returned to the Sequencer Page with the Edit Buffer intact.

DISCARD

If you choose **DISCARD**, the old data in the Edit Buffer is erased and the current sequence is copied into the Edit Buffer. You are moved into the Editor Pages with the current sequence displayed.

STORE

Choosing this option first clears the Edit Buffer by storing its contents to sequence memory, and then continues with the process of editing the current sequence as intended.

After you choose **STORE**, the Select Sequence window will appear asking “**STORE TO WHERE?**”. Scroll through the list with the cursor keys, or by dragging the scroll bar, to select the sequence that the Edit Buffer will be stored to. The default sequence is the one that the edit buffer was originally intended to replace (the target sequence at the time of recording), but you can choose any sequence. Just click **OK** to store the Edit Buffer.

Once the Edit Buffer is cleared, the current sequence will be copied into the Edit Buffer and you will be moved into the Editor Pages with the current sequence displayed.

EDIT IT

Choosing this option is like saying, “Edit the stuff that’s waiting in the buffer, instead of the sequence I was about to edit.” If you choose **EDIT IT**, you are moved into one of the Editor Pages with the old data still intact so you can continue editing it. The current sequence is left in sequence memory untouched.

DELETE

(SHIFT-“D” from the keyboard)

Directly below the **EDIT** button is the **DELETE** button. This button is used to erase the contents of the current sequence.

Select the sequence to be deleted, then click **DELETE**. A safety prompt will appear to prevent accidental erasure. Click **NO** to back out safely. Otherwise Click **OK** (or press the RETURN key) to erase the sequence.

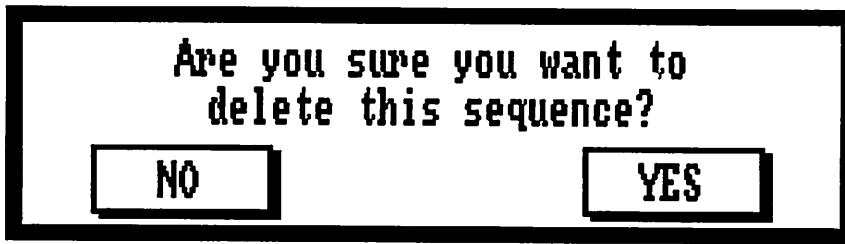


Illustration 6.20: Delete Safety Prompt

♦ Note: All sequences can be erased at once using the Options Menu / Erase All Sequences function.

[see Sequencer Page / Options Menu / Erase All Sequences]

PREVIEW

(SHIFT-“P” from the keyboard)

Directly below the **DELETE** button is the **PREVIEW** button. **PREVIEW** is used to substitute the contents of the Record Buffer for the current sequence during playback.

PREVIEW is turned on and off by clicking it. It's on when it's highlighted. **PREVIEW** causes Music-X to play the contents of the Record Buffer instead of the currently selected sequence. This is handy when trying different passes at a musical part; one can compare new and old versions before deciding. If the current sequence is being triggered by other sequences using Play Sequence events, then the Record Buffer will be substituted in all instances. (**PREVIEW** is temporarily ignored if changed during **PLAY** mode.)

Example: Using **PREVIEW** to compare new and old versions of a sequence.

- ☐ Record the first pass and **STORE** it in sequence 1.
- ☐ Record the second pass but don't **STORE** it. This leaves it in the Record Buffer.
- ☐ Turn **PREVIEW** on and make sure sequence 1 is the current sequence.
- ☐ Play back the piece. The old version (sequence 1) will be muted and the Record Buffer will be played in its stead.
- ☐ Turn **PREVIEW** off, relocate the clock to the beginning of the piece, and play the piece again. The old sequence will play and the Record Buffer will be silent.
- ☐ If you like the new version after previewing it, **STORE** the Record Buffer to sequence 1, writing it over the old version.

PREVIEW can be used after editing a sequence as well; simply choose **HOLD** when exiting the Editor Pages. The new version will stay in the Edit Buffer and can be previewed in place of the old version. (The Edit Buffer and the Record Buffer are the same thing, named differently depending on the situation.)

[see Bar Editor / File Menu / Exit / HOLD]



Chapter 7 - Bar Editor

Editing Concepts	126
Bar Editor Overview.....	128
Title Bar	129
File Menu	129
Edit menu	134
Options Menu	135
Display Menu	144
Modules Menu	146
Quantizer Module	146
Velocity Scaling Module	152
Aftertouch Scaling Module	157
Graphic Score Window	161
Event Types	166
Tool List	196
SetSig	196
Time Signature Display	197
Add	198
Insert	200
Move	201
Remove	202
Lock	202
Select	204
Mark	205

Bar Editor (continued)

Unmark	207
Zoom+ and Zoom-	207
Snap	208
Quiet	210
Scroll	211
Grid	212
Params	216
Repeat	220
Seq:	221
T=	222
D=	223
Transport Controls	223
Recording Modes	225
Channel Square	232
STEP and BACK	233
Virtual Sliders	234
Grid: and Dur:	235

Bar Editor

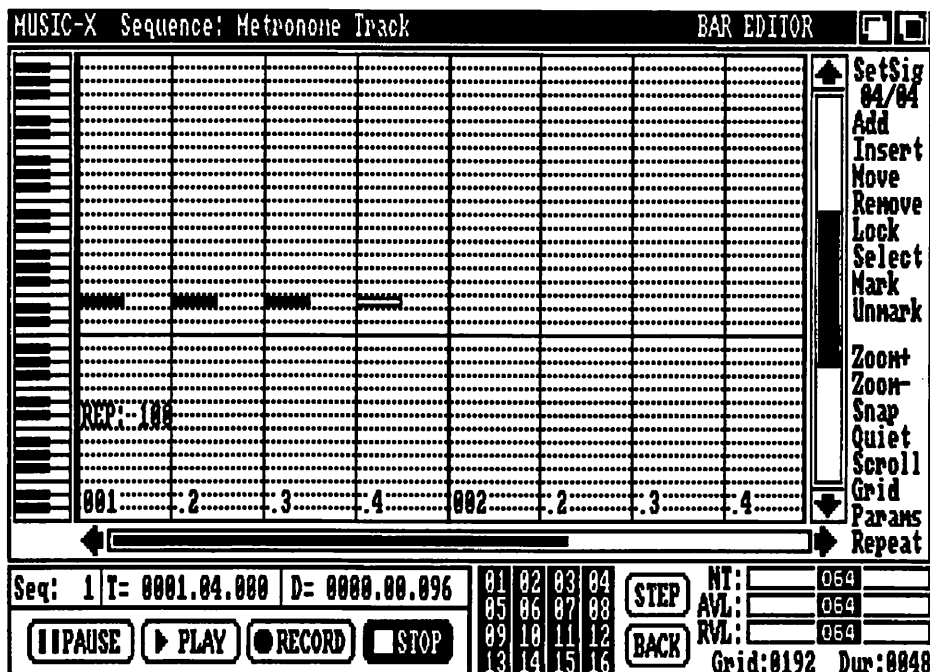


Illustration 7.1

Editing Concepts

What is an Editor?

An editor provides a way to look into a sequence and manipulate the contents; it lets you “get at the guts” of a sequence. An editor may be used to correct a prerecorded sequence, or to create a new sequence from scratch.

What is a Sequence?

A sequence is a list of things-to-do and when-to-do-them. These things-to-do are called “events”. Events play notes, send program changes, change the tempo, etc. When Music-X plays a sequence, it reads through the list and executes each event at the proper time.

Event Parameters

Each event can have multiple attributes or properties. These properties are called “parameters”. For example, a single note event has the parameters of start time, pitch, duration, MIDI channel, attack velocity, and release velocity.

Two Editors

Music-X has two sequence editors, the Bar Editor and the Event Editor. Each editor provides a different way to look at the same events. One can switch between the Bar Editor and the Event Editor freely, using each for its strengths.

Editing Single Events

Many events are displayed at once in the editor screens. In order to change the parameters of a single event, you first pick it out of the crowd. That event then becomes the “current event”. The controls in each editor are designed to automatically configure themselves to the parameters of the current event. To edit the parameters of a single event, you make it the current event and then tweak the controls.

Editing Multiple Events

It's possible to change the parameters of a group of events all at the same time. In order to do this the group must first be defined. There are two main ways to do this. The first way is to “select” the desired events individually until the group is complete. The second way is to “mark” a region of time. These methods are explained in more detail later. Once the group is defined, special editing functions can be called that operate on the group as a whole.

The Edit Buffer

When you first choose to edit a sequence, by clicking **EDIT** in the Sequencer Page, a copy of the current sequence is written into the Edit Buffer and you are moved into one of the editors. The editors are then used to make changes to the Edit Buffer without affecting the original sequence. It's safe to experiment. When satisfied with the changes, you store the newly edited version of the sequence back over the original.

◆ **Note:** The Edit Buffer is the same thing as the Record Buffer. In fact it's possible to edit the Record Buffer immediately after recording, without first storing it to sequence memory. Simply click **EDIT** from the sequencer page after recording, and when the safety prompt appears, click the **EDIT IT** option to edit the current contents of the Record Buffer.

[see Sequencer Page / EDIT]

Bar Editor Overview

In the Bar Editor, notes appear as colored bars—thus the name! (Not to be confused with “bar” meaning “measure”.)

The Bar Editor displays events graphically, as icons positioned in a two-dimensional time/pitch graph. Events can be added to, removed from, and repositioned in the graph using the mouse. This makes editing a sequence in the Bar Editor rather like using a paint program; notes can be “painted” onto the score using the mouse as a brush.

The rest of the Bar Editor chapter explains all the features of the Bar Editor Page one by one, roughly in the order that they're found on the screen (left to right, top to bottom).

Title Bar

The title bar is displayed along the top of all Music-X pages to show which page is currently being used. In the Bar Editor, the name of the sequence currently being edited appears here as well, following the word **Sequence:**.

The Title Bar becomes the Menu Bar, containing menu titles, when the right mouse button is pushed. To access any menu on any page, press the right mouse button and point to the desired menu title in the Menu Bar. The chosen menu will “pull down”, listing the available menu items. To choose a menu item, point to it, and release the right mouse button while the item is highlighted.

File Menu

The File menu is used to move whole sequences in and out of the Edit Buffer. Sequences can be moved to and from disk using **Save** and **Load**, or to and from sequence memory using **Store** and **Select**. This menu is available from both editors and functions identically in both cases.

Following are descriptions of the items available in the File menu, and the functions that they initiate.

Load...

(RightAmiga-“L” from the keyboard)

This function loads a previously saved sequence file into the Edit Buffer from disk so that it may be edited. The previous contents of the Edit Buffer are lost. Choosing **Load...** calls the File Requester. Use the File Requester to locate the sequence file on disk, then click **OK** to load the file. Otherwise, click **CANCEL** to return to the editors without loading.

[see File Requester chapter]

Save...

(RightAmiga-“S” from the keyboard)

This item saves the contents of the Edit Buffer to disk in a sequence file.

Choosing **Save...** calls the File Requester. Use the File Requester to locate the drawer to save your file in. Then use the File Requester to name your new file. The recommended file extension for sequence files is “.Seq”. Click **OK** to save the file, or click **CANCEL** to return to the editors without saving.

[see File Requester chapter]

Saving a sequence file also creates an icon for that file in the same drawer. When seen from the Workbench, the icons for sequence files look the same as performance file icons—like little audio cassettes.

◆ **Note:** A sequence file contains one sequence only. Multiple sequences may be saved in a performance file.

[see Sequencer Page / File Menu / Save Performance]

New

(RightAmiga-“N” from the keyboard)

This function erases the current contents of the Edit Buffer leaving a new blank sequence containing only an End event.

A blank sequence is a good starting place when building a sequence from scratch. Events can then be added one by one using the editor as a compositional tool.

Choosing **New** first calls a safety prompt to prevent accidental loss. Click **NO** to back out safely. Otherwise, click **YES** to proceed. The Edit Buffer is then erased.

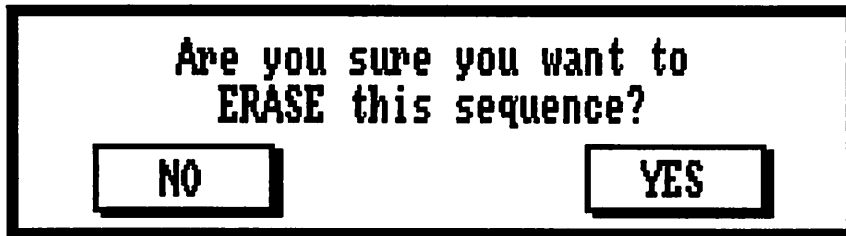


Illustration 7.2: New Safety Prompt

♦ Note: Using **New** does not change the current sequence number. That means that the default target when writing the Edit Buffer back to sequence memory will be the last sequence copied into the Edit Buffer before the **New** function is used. There are ways around this if your intention is to keep the old sequence as well. First, instead of using **New**, you can bring an empty sequence into the Edit Buffer using **Select...** (see below). Second, if you do use **New**, you can store the new contents to any of the 250 sequences in memory using **Store...** (see below).

Select...

(RightAmiga-"/" from the keyboard)

This function brings one of 250 sequences into the Edit Buffer from sequence memory. The previous contents of the Edit Buffer are lost.

Choosing **Select...** calls a window titled **SELECT SEQUENCE TO EDIT**. This window lists all of the available sequences in memory. The current sequence is marked by a dark highlight bar. Scroll through the list until the sequence you want to select is highlighted. Once a sequence is selected, click **OK** (or press the "O" key) to copy the sequence into the Edit Buffer. If you change your mind click **CANCEL**, or press the "C" key, to close the window without doing anything.

(CursorUp and CursorDown keys move through the list one entry at a time. SHIFT-CursorUp and SHIFT-CursorDown move through the list a page at a time.)

Store...

(RightAmiga-“T” from the keyboard)

This function writes the contents of the Edit Buffer back to one of the 250 sequences in memory. The previous contents of the target sequence are lost. You remain in the editor with the current contents.

Choosing **Store...** calls a window titled **SELECT SEQUENCE TO STORE TO**. This window lists all of the available sequences in memory. The current sequence is marked by a dark highlight bar. Scroll through the list until the sequence you want to select is highlighted. Once a sequence is selected, click **OK** (or press the “O” key) to write the Edit Buffer to the selected sequence. If you change your mind click **CANCEL**, or press the “C” key, to close the window without storing.

CursorUp and CursorDown keys move you through the list one entry at a time. SHIFT-CursorUp and SHIFT-CursorDown move you through the list a page at a time.

Recall

(RightAmiga-“R” from the keyboard)

This function recopies the current sequence into the Edit Buffer, restoring the Edit Buffer to the condition it was in when last stored. **Recall** can be used when you’re not satisfied with the changes made to a sequence so far, and you want to try again.

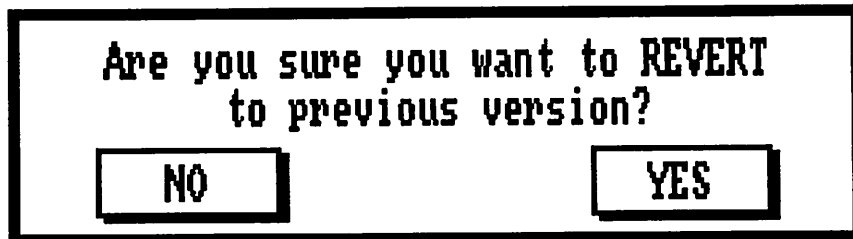


Illustration 7.3: Recall Safety Prompt

Choosing **Recall** first calls a safety prompt to prevent accidental loss. Click **NO** to

back out safely without recalling. Otherwise, click **YES** (or press RETURN) to proceed. The original sequence is then copied into the Edit Buffer.

Exit

(RightAmiga-“X” from the keyboard)

Choosing this menu item will move you out of the current editor and back to the Sequencer Page.

If no changes were made to the Edit Buffer while in the editor, you’ll be returned directly to the Sequencer Page. However, if any edits were made, a requester will appear asking what is to be done with the new data.

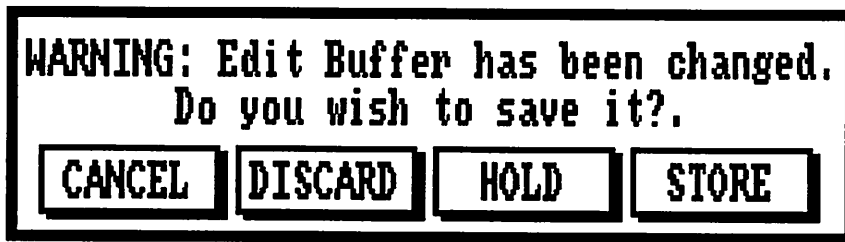


Illustration 7.4: Exit Safety Prompt

The choices offered by the requester, and their meanings, are listed below.

CANCEL

Click **CANCEL** if you change your mind about leaving the editor. The requester will go away, and you can continue editing.

DISCARD

Clicking **DISCARD** will lose any changes made in the editor before going back to the Sequencer Page. The Edit Buffer is erased and the current sequence is left in its original condition.

HOLD

Clicking **HOLD** causes the contents of the Edit Buffer to be held in the Edit Buffer without being erased or stored to sequence memory. You then move back to the Sequencer page.

There are two uses for **HOLD**. One is to temporarily leave the editors when you expect to come back and continue later. The other is to prepare for comparing new and old versions of a sequence using the **PREVIEW** button in the Sequencer Page.

[see Sequencer Page / PREVIEW]

STORE

Clicking **STORE** causes the contents of the Edit Buffer to be written to the current sequence before leaving the editors. The old version of the sequence is lost and the new version takes its place. The Edit Buffer is then erased and you are moved back to the Sequencer Page.

Edit Menu

This menu controls the overall editing environment. It is used to switch between the editor pages and to enable Auto-Step mode.

Following are descriptions of the items in the Edit Menu.

Bar

(RightAmiga-“B” from the keyboard)

Choosing this item moves you into the Bar Editor from the Event Editor.

Event

(RightAmiga-“E” from the keyboard)

Choosing this item moves you into the Event Editor from the Bar Editor.

Auto-Step

This menu item enables and disables Auto-Step mode for use when step recording. When recording normally, or when playing, Auto-Step is ignored.

Auto-Step is used, during step recording, to advance the clock automatically as keys are released on the mother keyboard. This is offered as an alternative to manually stepping the clock forward by clicking the **STEP** button. Step recording is described in more detail under Recording Modes.

[see Bar Editor / Recording Modes]

Auto-Step is enabled when this menu item is checked, and disabled when unchecked. The menu item may be turned on and off at any time, but is temporarily ignored if changed while recording.

Options Menu

The Options menu contains some of the extra editing options available in the editors. This menu is used to set the parameters that determine whether the sequence is to be played on the Amiga samples or through MIDI. The Options menu is also used to select the “cut and paste” type group editing operations involving the “Clip”.

The phrase “cut and paste” comes from the old days when manuscripts were edited, after being typewritten, by actually cutting up sheets of paper with individual paragraphs on them, and pasting them back together in different orders. This is also similar to the advertising term “paste up”, which refers to the process where the various pictures and text that make an advertisement are pasted onto one sheet before being photographed and reproduced. Nowadays, word processors are used to write and edit things like the manual you are reading, but the terminology for moving paragraphs around is the same.

In Music-X, the terms “Cut” and “Paste” mean basically the same thing as they did in the old days. You can “Cut” blocks out of a sequence, and “Paste” them back in at different places. This can be handy when doubling melodies, rearranging the order of sections within a sequence, etc.

Cutting and pasting is done using the “Clip”. The Clip is what holds the block that was cut out, before it gets pasted back in. The relationship each operation has with the Clip is explained below.

Following are descriptions of the items found in the Options menu.

Feedback

Feedback is what you hear when a Note event is dragged in the Bar Editor. The note is played at each new screen position to aid the ear when moving or placing notes in the score. This feedback can be played using Amiga samples or MIDI instruments.

◆ Note: Feedback can be turned off completely by clicking **Quiet** in the Tool List along the right edge of the Bar Editor Screen. Feedback is automatically turned off when the **PAUSE** button is on.

[see Bar Editor / Quiet]

Feedback has a pop-out menu with two items. The currently checked item shows how feedback will be played. The **Feedback** setting is reset when a new sequence is brought into the editor and defaults to the setting of the **Out** field of the current sequence. The choices are listed below.

[see Sequencer Page / Sequence List / Out]

MIDI

When **MIDI** is checked, all feedback will be played on external MIDI instruments. (A MIDI instrument must be connected for feedback to be heard.)

Amiga

When **Amiga** is checked, all feedback will be played internally on Amiga samples. (There must be samples currently in memory for feedback to be heard.)

[see Amiga Samples chapter]

Output

This item controls the same parameter as the **Out** field in the Sequence List on the Sequencer Page. Changing the parameter here also changes it in the Sequencer Page. It determines to which output bank the sequence will be sent when played back from the editors or the Sequencer Page. A pop-out menu contains two items. The currently checked item shows the current output bank. These choices are listed below.

[see Sequencer Page / Sequence List / Out]

MIDI

When **MIDI** is checked, the current sequence will be played using external MIDI instruments. (MIDI instruments must be connected for the sequence to be heard.)

Amiga

When **Amiga** is checked, the current sequence will be played using internal Amiga samples. (There must be samples currently in memory for the sequence to be heard.)

Cut

(RightAmiga-“K” from the keyboard, think “Kut”)

Cut is used to remove a block of data from the sequence and put it into the Clip.

Before **Cut** can be used, a block of data must be defined. This is done using the tool words **Select**, or **Mark**. With **Select**, you define a block of data by selecting all the events you want to cut, one by one, until the block is complete. With **Mark**, you define the block by defining a region of time in the sequence. **Cut** works differently, depending on the type of block that has been predefined.

[see Bar Editor / Tool List]

[see Bar Editor / Select]

[see Bar Editor / Mark]

Choosing **Cut**, after predefining a group of events with **Select**, copies those events into the Clip (overwriting its previous contents), then removes them from the sequence. No other events in the sequence are altered. The next logical step after Cutting a group of events out of the score is to paste them back in at another location. To facilitate this, and to save a step, the editor moves directly into "Paste mode" (see **Paste** below) after a **Cut** operation. The events in the Clip appear attached to the mouse pointer, and the pointer changes to a hollow "X" (as in "X marks the spot"). If you decide against pasting immediately, click the right mouse button once to exit Paste mode and return to normal. Otherwise, by clicking anywhere in the score window, you can paste the Clip into the sequence at that location.

Choosing **Cut**, after predefining a region of time with **Mark**, copies that region of time into the Clip (overwriting the previous contents), then removes that region of time from the sequence. All events after the end of the region are moved forward (earlier) to fill the gap. (This is analogous to cutting out a piece of audio tape and splicing the two loose ends together.) The mouse pointer changes to a hollow "X", a thick vertical green line appears attached to the mouse pointer, and the editor moves into **Paste** mode (see below). If you decide against pasting immediately, click the right mouse button once to exit **Paste** mode and return to normal. Otherwise, by clicking anywhere in the score window, you can insert the Clip into the sequence at a new location, moving any following events back (later in time) to make room for the Clip.

Choosing **Cut** while only the current event is selected, copies the current event into the Clip and removes it from the sequence. The editor then moves into **Paste** mode.

Choosing **Cut**, while there are currently no selected events or marked time will do nothing.

◆ **Note:** The Clip will hold its contents until new contents are put into it. Exiting the editors, or bringing new sequences into the editors, will not cause it to lose its contents. This means the Clip can also be used to exchange blocks between sequences.

Copy

(RightAmiga-“C” from the keyboard)

Copy is similar to **Cut**, in that it takes a predefined block and puts it into the Clip. The difference is that **Copy** does not remove the block from the sequence, and the sequence is left intact. **Copy** works differently, depending on the type of block that has been predefined (Selected events, or Marked time).

Choosing **Copy** after selecting a group of events, copies those selected events into the Clip, overwriting the previous contents, then deselects them. The Clip then appears attached to the mouse pointer, the pointer changes, and the Editor moves directly into **Paste** mode (see **Paste** below). If you decide against pasting immediately, click the right mouse button once to exit **Paste** mode and return to normal. Otherwise, by clicking anywhere in the score window, you can paste the Clip into the sequence at that location.

Choosing **Copy** while there is a marked region of time copies that region of time into the Clip, overwriting the previous contents, and unmarks the region. The green marker then appears attached to the mouse pointer and the Editor moves directly into **Paste** mode. If you decide against pasting immediately, click the right mouse

button once to exit **Paste** mode and return to normal. Otherwise, by clicking anywhere in the score window, you can paste the Clip into the sequence at that location.

Choosing **Copy** while only the current event is selected will copy the current event into the the Clip.

Choosing **Copy** while there are currently no selected events or marked time will do nothing.

Delete

(RightAmiga-“D” from the keyboard)

Delete is used to remove and discard a predefined block from the sequence. **Delete** works in different ways depending on the type of block being deleted. **Delete** does not actually involve the Clip.

Choosing **Delete** after Selecting a group of events, removes those events from the score. No other events are altered. The contents of the Clip are not affected. The deleted events are lost.

Choosing **Delete** after Marking a region of time, removes that region of time from the sequence. All following events are moved forward (earlier) to fill the gap. The contents of the Clip are not affected. The deleted time is lost.

Choosing **Delete** while only the current event is selected removes and discards the current event. No other events are altered. The contents of the Clip are not affected.

◆ **Note:** If you make a mistake and delete the wrong data by accident, remember that **Recall** can be used to restore the sequence to the condition it was in when it was last stored.

[see Bar Editor / File Menu / Recall]

Paste

(RightAmiga-“P” from the keyboard)
(SHIFT key limits transposition)

Paste is used to place the current contents of the Clip into the sequence at any desired time point, and at any transposition. The Clip is not erased by this action, so multiple copies of the Clip may be repeatedly pasted into the score. For example, small melodic fragments could be pasted in clusters to build more complex patterns, or a single horn voicing could be pasted repeatedly under each note in a melody to create parallel harmonies. **Paste** works differently, depending on the type of block that is currently in the Clip (Selected events, or Marked time).

When the Clip contains a group of selected events, those events are added to the sequence without affecting any previous events in the sequence. Choosing **Paste** first changes the mouse pointer to a hollow “X”. The events in the Clip then appear as a cluster, in green, attached to the mouse pointer, with the first event in line with the center of the “X”. Moving the mouse pointer moves the cluster around in the score window. Moving the cluster horizontally repositions it in time. The **T=** indicator below the score window shows the start time of the first event in the cluster. Moving the cluster vertically changes the transposition of the events in the cluster. The **D=** indicator below the score window is used to show the transposition factor, read as a halfstep distance from the original key. (Example: “-1” means one halfstep down. “0” means no transposition. “1” means one halfstep up.) The transposition of the Clip can be limited to zero by holding the SHIFT key while placing the cluster into the score. If you decide not to add the Clip to the sequence, click the right mouse button. The pointer will return to normal, and the event cluster will disappear. Otherwise, once you’ve positioned the Clip, press the left mouse button. All the events in the Clip will be added to the sequence at their current positions. Pasting automatically deselects any previously selected events in the score, and selects all the newly added events. This allows you to further edit the new events after they are added. You may, for instance, change all their durations with the CursorLeft and CursorRight keys, or change their MIDI channels with the Channel Square, or even delete them once again with the **Delete** option in this

menu. If you are satisfied with the new events as they are, click **Unmark** in the Tool List along the right edge of the Bar Editor to deselect the new events and set them into the sequence in their current state.

When the Clip contains a region of time, that time is inserted into the score. Events after the insertion point are pushed back (later) in time to make room for the new region. Choosing **Paste** first changes the mouse pointer to a hollow “X”. A vertical green line then appears attached to the pointer, passing through the center of the “X”. This line marks the insertion point. Moving the pointer horizontally moves the insertion point through time. The **T=** indicator below the score window shows the position of the insertion point as it changes. Moving the pointer vertically changes the transposition for the events in the Clip that will be inserted with the region of time. The **D=** indicator, below the score window, is temporarily used to show the transposition factor. If you decide not to paste the Clip into the score, click the mouse pointer anywhere outside of the score window. The pointer will return to normal, and the green line will disappear. Otherwise, once you have positioned the insertion point, with the desired transposition, press the left mouse button. The region of time will then be inserted into the sequence at that point. The new region will stay **Marked** (the region will appear highlighted in purple) after being inserted, to allow for further editing. You may, for instance, change the transpositions of the events within the region by pressing the **CursorUp** and **CursorDown** keys, or change their **MIDI** channels with the **Channel Square**. If you are satisfied with the new range as it is, click **Unmark** to Unmark the region and set it into the sequence in its current state.

Select All

(RightAmiga-“A” from the keyboard)

Choosing this item selects all *currently displayed* events in the sequence. Events turned off in the Display Menu are not selected. **Select All** can be used in preparation of one of the other group editing procedures such as **Copy** or **Delete**. Example: Removing Channel Aftertouch events from a sequence.

- ☐ From the **Display** menu, turn off everything except Channel Aftertouch events. That means turn off **Notes**, **Program Change**, **Pitch Bend**, and **System Exclusive**.
- ☐ From the **Display** menu, turn on **Channel Aftertouch**. Nothing should now be displayed except Channel Aftertouch events.
- ☐ Choose **Select All** from the options menu. All the Channel Aftertouch events are now selected.
- ☐ From the **Options** menu, choose **Delete**. This removes the selected events from the sequence.
- ☐ From the **Display** menu, turn everything back on.

[see Bar Editor / Display Menu]

[see Bar Editor / Select]

◆ **Note:** Not all event types can be displayed at the same time. Therefore **Select All** cannot be easily used to do operations on the whole sequence. Use **Mark** instead to define a region of time that includes the whole sequence.

Show Clip

When **Show Clip** is chosen, the contents of the Clip are shown in the score window instead of the current sequence. This can be used to verify the composition of the Clip before pasting.

While the Clip is being shown, the score window may be zoomed, scrolled, and even played. **Display** menu settings may be changed as well.

To bring the score window back to normal (showing the contents of the Edit Buffer), click once, anywhere within the window.

Display Menu

The **Display** menu is used to control which event types can be seen in the graphic score window. By selectively turning off some events, other events can more easily be edited. Events not displayed can still be heard when the sequence is played.

Nine event types are listed in the Display menu. Each event type is turned off and on by selecting it. Menu items for currently enabled event types will appear checked.

◆ **Note:** Adding or Inserting a new event to the score automatically enables the displaying of that event type, and disables the displaying of any similar looking events, so they can be distinguished.

[see Bar Editor / Add]

[see Bar Editor / Insert]

◆ **Note:** Events can also be selectively hidden according to MIDI channel using the **Lock** tool.

[see Bar Editor / Lock]

Following are descriptions of the items in the **Display** menu.

Notes

When this item is checked, Note events will be displayed. Note events appear as horizontal bars. Notes take on the color of the MIDI channel they're on.

Attack Velocity

When this item is checked, the attack velocities of Note events will be displayed. Attack velocities appear as separate dark blue rectangles extending upward from the bottom of the score window, placed directly beneath their associated Note events. The height of each rectangle shows the intensity of the Note's attack velocity.

When Note events are turned off, attack velocities are not displayed. (Turning velocities off when not needed can speed up the display a bit when scrolling.)

Release Velocity

When this item is checked, the release velocities of Note events will be displayed. Release velocities look the same as attack velocities. One velocity type can be shown at a time. When **Release Velocity** is enabled, **Attack Velocity** is automatically disabled.

When Note events are turned off, release velocities are not displayed. (Turning velocities off when not needed can speed up the display a bit when scrolling.)

Program Change

When this item is checked, Program Change events will be displayed. Program Change events appear as text written in the score, colored according to their MIDI channel. The abbreviation “PGM:” is followed by the program number. Example: **PGM: 090**.

Channel A.T.

Read this menu item as “Channel Aftertouch”.

When this item is checked, Channel Aftertouch events are displayed. Aftertouch events look rather like pins; they are displayed as dots supported by lines connected to the bottom of the score window, colored according to their MIDI channel. The height of each dot shows the relative aftertouch pressure.

Polyphonic A.T.

Read this menu item as “Polyphonic Aftertouch”.

When this item is checked, Polyphonic Aftertouch events will be displayed. Both types of aftertouch, as well as Control Change events, look the same. When **Polyphonic A.T.** is turned on, the other two are turned off.

Control Change

When this item is checked, Control Change events will be displayed. Control Change events are displayed the same way as the two Aftertouch types. When any of the three are turned on, the other two are automatically turned off.

Pitch Bend

When this item is checked, Pitch Bend events will be displayed. Pitch Bend events appear as small diamonds colored according to their MIDI channel.

System Exclusive

When this item is checked, System Exclusive events will be displayed. System Exclusive events appear as small black vertical dashes spanning the height of about two pitch lines.

Modules Menu

The **Modules** menu is where access is given to externally loaded editing modules. These modules are small programs that operate on the data in the Edit Buffer in special ways. They can be called and used from either editor. Three editing modules are included with Music-X as released. The menu items that call them are:

Quantize...

Scale Velocity...

Scale Aftertouch...

The following three subchapters describe these Modules.

Quantizer Module

To call this module from within either editor, choose **Quantize...** from the Modules menu. The Quantizer is used to alter (correct) the timing of events in a sequence.

The Quantizer is so named because it deals with quantities of time. Most music assumes an underlying regular pulse on which the music is placed. This is represented in Music-X by the "Grid". The Quantizer is used to make sure the timed execution of notes is nice and close to this ideal Grid. A quantized sequence will sound very even and regular.

The spacing between Grid points is called the "Grid Size". By using different Grid sizes, you can get triplets, etc.

The Grid Size is set using the **Grid** tool found in the Tool List on the right edge of the Bar Editor Page. This should be done before calling the Quantizer, to set up the appropriate Grid and Duration. The currently selected Grid Size gives an idealized timing reference used as a target by the Quantizer when moving the start times of events. Duration can be explained as the length of time between the start and the end of a note (between the MIDI Note On and Note Off). The currently selected duration is used by the Quantizer as an idealized reference when quantizing durations.

[see Bar Editor / Grid]

The Error Spectrum

Let's say we've set up a Grid of quarter notes. Each Grid point is 192 clocks away from the next. (A clock is the smallest amount of time possible in a Relative

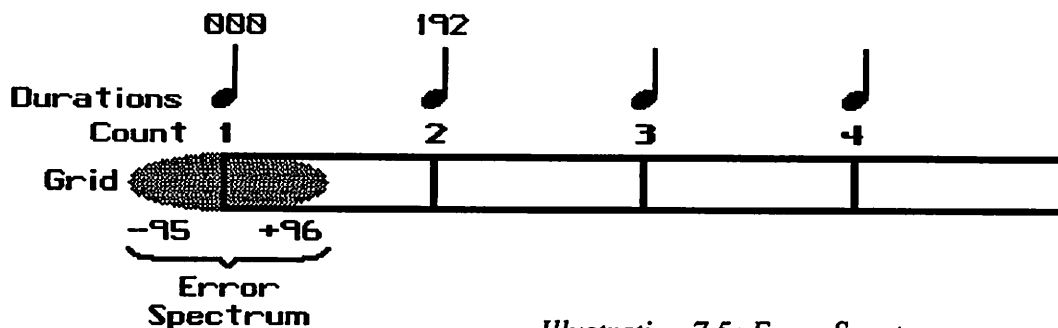


Illustration 7.5: Error Spectrum

sequence.) These are our ideal “downbeats”. Now imagine the area of time surrounding each Grid point and extending halfway to the next Grid point in both directions (earlier and later). This area of time creates a margin around each Grid point that can be called the “error spectrum”. Any notes within the error spectrum are considered by the Quantizer to be deviations from the ideal Grid point in the center of the spectrum. The further away from the center that a note starts, the greater the timing error. So, for instance, with our Grid of quarter notes, which is 192 clocks wide, a note may start as early as 95 clocks before the beat or as late as 96 clocks after the beat and still be quantized correctly.

After you’ve recorded a live performance, chances are that there will be minor (or major) differences in timing between the way you played it and the way you wish you’d played it. The Quantizer will “comb” through all the notes you played and move them to the closest Grid point. Say you played a four-note arpeggio, approximately in quarter notes. And let’s assume that we’ve previously set a Grid of quarter notes (192 clocks) by using the Grid requester. Here are the times before and after quantization (shown as measures.beats.clocks).

<u>Note</u>	<u>Played time</u>	<u>Quantized time</u>
C4	0001.01.010	0001.01.000
E4	0001.02.004	0001.02.000
G4	0001.02.190	0001.03.000
C5	0001.03.165	0001.04.000

Knowing that each quarter note contains 192 clocks, you can see here that the first note was played late by 10 clocks, the second was late by 4 clocks, the third note was early by 2 clocks and the fourth note was early by 27 clocks. The Quantizer moved each one to the closest nearby Grid point.

In the right half of the Quantizer Module's window is a column of buttons for the various quantizing options. Following are descriptions of those options.

QUANTIZE EVENTS		Quantization Size: 0192	
Min Threshold: 00		☐	Start Only
Max Threshold: 96		☐	Duration Only
Effect %: 100		☐	Stop Only
☒ Selected Events	☐ All	☐	Start + Duration
		☐	Start + Stop
QUANTIZE CANCEL		☒	Start w/Duration

Illustration 7.6

START ONLY will move only the Note On portion of Note events to the nearest Grid point and leave the Note Offs where they were. The durations may be arbitrarily changed.

DURATION ONLY will leave the Note Ons where they are and move each Note Off earlier or later in time to cause the duration of the note to resolve to the nearest multiple of the chosen duration. Example: If you have chosen eighth notes as your duration, and the original duration of a note to be quantized is somewhere between a true quarter note (two eighths) and a true dotted quarter note (three eighths), then the Quantizer will move the Note Off earlier or later to whichever value is closer; the note will end up being exactly a quarter note or exactly a dotted quarter note in duration.

STOP ONLY will move only Note Offs to the closest Grid point. Note Ons are not moved. Durations may be arbitrarily changed.

START + DURATION will move the Note Ons towards the nearest Grid point and adjust the Note Offs so that the duration is moved towards the nearest multiple of the chosen duration. In most cases this is identical to **START + STOP** (see below). It would be different in cases where the Grid Size and the chosen Duration are different. Suppose you set your Grid to sixteenth notes and your Duration to whole notes. The Note Ons would be moved to the closest Grid point but instead of adjusting the end points to the nearest sixteenth note Grid, the Quantizer would adjust the end points to make the over-all duration of the note some multiple of whole notes. This is a versatile feature. Experiment with different settings.

START + STOP will move both Note Ons and Note Offs towards the nearest Grid point (even in opposite directions). Durations may be altered arbitrarily, but will end up as some multiple of the Grid Size (not necessarily some multiple of the current Duration setting).

START w/DURATION means “start with same duration”. In other words, “Quantize the start time but leave the duration the same”. This will move the Note Ons to the nearest Grid point and move the related Note Off in the same direction by the same amount, leaving the over-all duration of the note the same as it was originally played. This is good for correcting the timing of a phrase while staying close to the original articulation. The gaps between notes retain most of their original feel; legato or staccato sections will still sound legato or staccato.

In the left half of the Quantizer Module’s window are three virtual sliders that limit the overall effect of quantization.

Minimum Threshold: This slider tells the Quantizer at what point along the error spectrum to start quantizing. You can choose NOT to quantize rhythms that are within a certain number of clocks from the Grid center, early or late. This is like saying, “If it’s close enough already, let’s leave it alone so it doesn’t sound too robotic!” You can set the threshold from 000 clocks to 1/2 the current Grid Size or Duration, whichever is bigger. Example: 000 to 96 clocks are available when

quantizing quarter notes (a quarter note is 192 clocks). Setting a minimum threshold of 5 will only quantize notes occurring at 5 clocks or further from the Grid center.

Maximum Threshold: This slider tells the Quantizer at what point along the error spectrum to stop quantizing. You can choose to leave rhythms as they are if they are beyond a certain number of clocks from the Grid, early or late. This is like saying, “If it’s THAT far off, I must have meant it!” Possible settings range from 000 clocks to 1/2 the current duration or current Grid Size, whichever is bigger. Example: 000 to 24 clocks are available when quantizing sixteenth notes (a sixteenth note is 48 clocks). Setting a maximum threshold of 10 would only quantize notes occurring at 10 clocks or closer from the Grid center.

Effect %: Read this slider as “Effect Percent”. This slider lets you tell the Quantizer to only partially quantize the notes; to just move the notes part of the way towards the nearest Grid or Duration. If your note is 10 clicks away from the Grid and you choose a 50 percent effect, the note will be moved to 5 clicks away from the Grid. If you still want it closer, run the Quantizer again, the note will be moved a little closer each time. It’s like saying, “The track’s pretty close as it is. I just want it a little tighter without making it sound stiff”. If you are quantizing Durations, then the effect percent slider will partially quantize them as well.

Below the sliders are two switches that determine which overall group of events will be considered for quantization.

Selected Events When this switch is on, only those events currently Selected in the sequence, or those events within a Marked region of time will be quantized according to the parameters set above. If no events are currently Selected, or Marked, and this switch is on, then *no effect will be heard*. When the **Selected Events** switch is on, the **All** switch (see below) is automatically turned off.

[see Bar Editor / Select]

[see Bar Editor / Mark]

All When this switch is on, all the events in the current sequence will be quantized according to the parameters set above. Actually, only Note events and certain Music-X events will be affected. Everything else is left alone automatically. It's assumed, for instance, that nobody wants to quantize pitch bend information.

[see Bar Editor / Event Types]

QUANTIZE This button, when clicked, causes the Quantizer to go ahead and quantize according to the parameters set up in the other controls. When done setting up the parameters, click **CANCEL** or press the "Q" key to effect the quantization.

CANCEL This button can be used to safely close the Quantizer window without changing any sequence data. Click **CANCEL** or press the "C" key to safely exit the Quantizer.

Velocity Scaling Module

This Module can be called and used from either editor by choosing **Scale Velocity...** from the Modules menu.

The Velocity Scaling module is used to alter the attack and release velocities of notes in a sequence. It can operate on the whole sequence or on any set of two or more notes within the sequence.

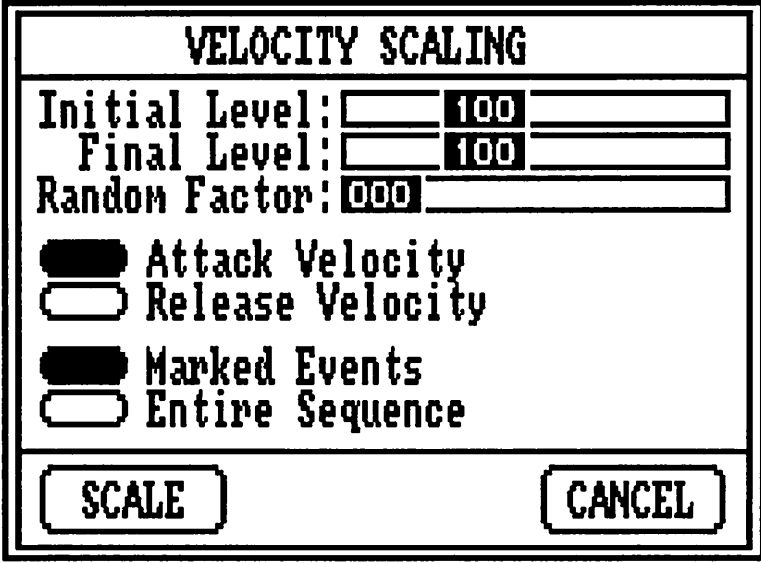
What is velocity?

Attack velocity and release velocity are two of the parameters associated with a Note event and are sent in MIDI Note On and Note Off messages respectively.

The attack velocity of a note relates to how fast the key was pressed on the mother keyboard. (This does not mean a faster musical tempo, but rather how fast the key travels from full up to full down position. Most performers think of this as how "hard" they hit the keys.) Various synths use this parameter to control either the

timbre of the sound, the volume of the sound, or both. Some synths use it to control the attack rate or other portions of their envelope generators as well. The theory is that if you play faster, you want the sound louder, brighter, and quicker in responding. And if you play slower, you want it softer, darker, and slower in responding.

The release velocity of a note relates to how fast you release the key from the depressed position. Like attack velocity, this is used by various synths to control the timbre, volume, or articulation of the sound. Unfortunately, it's less widely supported by synth manufacturers than attack velocity is. In the hardware of an electronic keyboard, velocity is actually measured by causing a key to close or open two separate switches, one near the beginning of travel and one near the end, and noting the amount of time between the closings. It really has nothing to do with how hard you play the key but rather, how fast.



VELOCITY SCALING

Initial Level: 100

Final Level: 100

Random Factor: 000

☒ Attack Velocity

☐ Release Velocity

☒ Marked Events

☐ Entire Sequence

SCALE **CANCEL**

Illustration 7.7

Choosing **Scale Velocity...** from the Modules menu opens the **VELOCITY SCALING** window. The scaling module affects the intensity of attack and release

velocities of Note events according to parameters set up in the window by you. It's generally used to create crescendos (soft at the beginning, loud at the end) or decrescendos (loud at the beginning, soft at the end), or to raise or lower the velocity of a range of notes all together.

Following are descriptions of the parameters in the **VELOCITY SCALING** window.

Initial Level:

The Initial Level slider sets the desired percentage of current velocity for the first (initial) note in a range.

This parameter together with the **Final Level** parameter creates a ramp of percentages over time from the initial event to the final event.

The slider ranges from 000 percent of current velocity to 300 percent of current velocity. As each event is processed, its velocity will be changed by the percentage dictated by the level of the ramp at that point in time.

Final Level:

The Final Level slider sets the desired percentage of current velocity for the last (final) note in a range.

If the Final level is set higher than the Initial level, then a crescendo-type effect will be imparted.

If the Final level is set lower than the Initial level, then a decrescendo-type effect will be imparted.

If the Final Level is the same as the Initial Level, then the ramp will be flat. The effect will be one of an overall increase or decrease in dynamics.

If both the Initial Level and the Final Level are set at 100 percent, then no ramp is set up and no changes will be made to velocities by these parameters. (However, the Random Factor parameter may still be live.)

Random Factor:

You can choose to have the velocities of Note events changed by adding a random offset to each. This is a simple addition and has nothing to do with percentages.

The slider sets the width of the range of possible offsets. A width of 100 defines a range from -50 to +50 to be added to the current velocity.

Example: If a velocity is 100, and the slider is at 100, then the possible new velocities are 50 through 150.

Try this on a constant-velocity high-hat part to hear what happens. Or do it to keyboard parts entered with a non-velocity-transmitting mother keyboard to add some life (your sound module must support velocity sensitivity for you to hear the effect).

Attack Velocity

When this switch is on (highlighted), the attack velocities of affected Note events will be scaled.

Release Velocity

When this switch is on, the release velocities of affected Note events will be scaled. If this switch is off and the **Attack Velocity** switch is off as well, then of course nothing will happen.

Marked Events

When this switch is highlighted, only Note events that are currently Selected, or are

within a Marked time region, will be affected by the scaling operation. When this is on, the **Entire Sequence** option (see below) is turned off.

[see Bar Editor / Select]

[see Bar Editor / Mark]

Entire Sequence

When this is highlighted, all the Note events in the sequence will be affected by the scale operation (including any events after the end line). When this is on, the **Marked Events** option is turned off.

SCALE

Clicking the **SCALE** button tells the software to go ahead and do the scaling operation using the parameters that have been set in this window.

Interesting effects can be accomplished by changing the ramp and running the scaler again over the same section of a sequence. Complex dynamic shapes can also be created by giving different sections individual crescendos and decrescendos.

◆ Note: The possible complexity of combined scale operations means scaling can't be done by simple offset during playback; therefore, the original data must be altered. Make sure you like the result before leaving the editor and saving your changes.

CANCEL

Clicking **CANCEL** will Scale nothing, close the Velocity Scaling window, and return to the editor.

Aftertouch Scaling Module

This Module can be called and used from either editor by choosing **Scale Aftertouch** from the **Modules** menu.

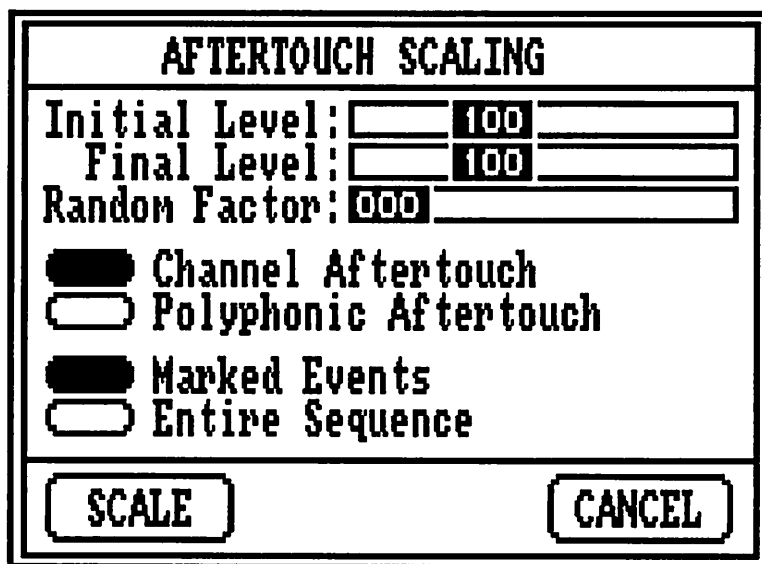
The Aftertouch Scaling module is used to alter the pressure values of Aftertouch events in a sequence. It can operate on all the events in a sequence or on any set of two or more Aftertouch events within the sequence.

What is Aftertouch?

Aftertouch messages relate to how hard a key is pressed on the mother keyboard after initially reaching the bottom of travel. Aftertouch provides an extra means of expression after the note has been initially struck. Some wind controllers send Aftertouch messages when breath pressure is increased during a note. Various synths and sound modules use Aftertouch to affect the timbre, volume, or pitch of the sound. Generally, the harder you press, the “brighter” the sound gets, and possibly the louder. Not all keyboards send or respond to Aftertouch events. To hear the effects of the Aftertouch Scaling module, your sound source must support Aftertouch.

There are two kinds of Aftertouch events, one that affects an entire MIDI channel, and one that affects individual notes. The first is called “Channel Aftertouch”, and the second, “Polyphonic Aftertouch”. Mother keyboards that send Channel Aftertouch messages send messages containing a value that represents the overall pressure being exerted across their keyboard. Mother keyboards that support Polyphonic Aftertouch can sense pressures being applied to individual keys. Owners of mother keyboards that support Channel Aftertouch will find that varying pressures applied to a single note in the right hand will also affect the notes of a chord being held with the left hand. Owners of mother keyboards that support Polyphonic Aftertouch will find that each note in a chord can be affected separately.

Choosing **Scale Aftertouch...** from the Modules menu opens the **AFTERTOUCH SCALING** window. This window is used to set the parameters that shape the way the module will alter the pressure values of Aftertouch events over time. The parameters in the **AFTERTOUCH SCALING** window are similar to the parameters in the **VELOCITY SCALING** window. Following are descriptions of those parameters.



AFTERTOUCH SCALING

Initial Level: 100

Final Level: 100

Random Factor: 000

☒ Channel Aftertouch

☐ Polyphonic Aftertouch

☒ Marked Events

☐ Entire Sequence

SCALE **CANCEL**

Illustration 7.8

Initial Level:

The Initial Level slider sets the desired percentage of current pressure for the first (initial) Aftertouch event in a range.

This parameter together with the **Final Level** parameter creates a ramp of percentages over time from the initial event to the final event.

The slider ranges from 000 percent of current pressure to 300 percent of current pressure. As each Aftertouch event is processed, its pressure will be changed by the percentage dictated by the level of the ramp at that point in time.

Final Level:

The Final Level slider sets the desired percentage of current Aftertouch for the last (final) Aftertouch event in a range.

If the Final level is set higher than the Initial level, then a crescendo-type effect will be imparted.

If the Final level is set lower than the Initial level, then a decrescendo-type effect will be imparted.

If the Final Level is the same as the Initial Level, then the ramp will be flat. The effect will be one of an overall increase or decrease in dynamics.

If both the Initial Level and the Final Level are set at 100 percent, then no changes will be made to pressures by the ramp parameters. (However, the Random Factor parameter may still be live.)

Random Factor:

This slider sets the width of a random offset to be added to the pressure values of Aftertouch events. The slider ranges from 0 to 100.

A width of 100 defines a range from -50 to +50 to be added to the current Aftertouch.

Example: If the pressure of an Aftertouch event is at 100, and the slider is at 100, then the possible new velocities are 50 through 150.

Channel Aftertouch

When this switch is on (highlighted in purple), Channel Aftertouch events will be affected by the scaling operation.

Polyphonic Aftertouch

When this switch is on (highlighted in purple), Polyphonic Aftertouch events will be affected by the scaling operation.

◆ Note: The **Channel Aftertouch** and **Polyphonic Aftertouch** switches are independent and may be turned on or off separately. If both are on, then both types of events will be affected by the scaling operation. If both are off then the Aftertouch Scaling module can have no effect.

Marked Events

When this switch is highlighted, only Aftertouch events that are currently Selected, or are within a Marked time region will be affected by the scaling operation. When this is on, the **Entire Sequence** option is automatically turned off.

[see Bar Editor / Select]

[see Bar Editor / Mark]

Entire Sequence

When this is highlighted, all the Aftertouch events in the sequence will be affected by the scale operation. When this is on, the **Marked Events** option is automatically turned off.

◆ Note: When using **Entire Sequence**, the ramp ranges from the beginning of the sequence to the last event in the sequence (even if after the END line).

SCALE

Clicking the **SCALE** button closes the window and initiates the scaling operation using the parameters you have set up.

CANCEL

Clicking **CANCEL** will close window without scaling anything, and return you to the editor.

Graphic Score Window

The graphic score window is the most prominent feature of the Bar Editor, found just below the title bar and occupying most of the screen. This is the work area, where the sequence is displayed and edited.

The graphic score window is basically a two dimensional time/pitch graph. Time moves horizontally, left to right; pitch is vertical, with low notes at the bottom and high notes at the top. The graphic score window scrolls to reveal the changing portions of the sequence being played. New events enter from the right and are seen and then heard as the sequence plays.

◆ Note: Auto-scrolling can be turned off and on by clicking **Scroll** along the right edge of the Bar Editor Page.

[see Bar Editor / Scroll]

Following are descriptions of the features of the graphic score window.

Keyboard

Along the left edge of the window is a miniature graphic of a piano keyboard. This is used to show the correspondence between note events in the score window and normal keyboard notes.

Measure Lines

Vertical blue lines are used as a background in the score window to show how musical time is mapped in the sequence. A dark line marks the beginning of each measure, and lighter lines mark the beats within that measure. The number of beats within a measure will depend on the current time signature. As Time Signature

Change events are played, the measure and beat lines are redrawn to match the new time signature.

[see Bar Editor / Event Types / Time Signature Change]

When editing Absolute sequences, the measure lines are used to show clock time instead of musical time. Light blue lines are spaced at one second intervals.

Measure Numbers

Measure and beat numbers are written in white along the bottom of the graphic display window. They correspond to the measure and beat lines described above. Measure numbers always have four digits with no decimal point. Beat numbers always start with a decimal point and are two digits long.

When editing Absolute sequences, the measure numbers are replaced by a single clock number written in the same format as the real time clock in the Sequencer Page. This number indicates the starting time for the current portion of the sequence, starting at the left edge of the score window.

Pitch Lines

Horizontal pitch lines are drawn on the background of the score window to show the possible positions for note events. The higher the position, the higher the MIDI key number of that note, and the higher the pitch.

The pitch lines extend from the left edge of the score window and can be matched against the mini keyboard. The lines correspond to the cracks between the white keys of the keyboard. The spaces between the lines correspond to the white keys.

The line below middle C is drawn in a darker blue for reference.

Time Line

When you click **PLAY** or **RECORD** in the Bar Editor, you'll see a moving vertical dotted green line cross the screen. This is the Time line.

The Time line shows where musical time is, in relation to the events in the graphic display. You'll hear the events as this line passes over them. The line starts from the left edge of the screen and moves to the right. When the line reaches the center, the score starts moving under it (assuming **Scroll** hasn't been turned off). When you click **PAUSE**, or **STOP**, the time line will stay in its last position, making it useful for locating mistakes by ear and eye.

[see Bar Editor / Scroll]

Octave Scroll Bar

MIDI supports 128 possible key numbers. That's over ten octaves. Five octaves (60 keys) can be shown in the graphic score window at one time. Use the scroll bar on the right side of the graphic score window to scroll the window vertically, bringing the outlying octaves into screen view.

Time Scroll Bar

Use this scroll bar, found at the bottom edge of the graphic score window, to relocate the window to different parts of the sequence. The scroll bar may be used at any time, except when **Scroll** is on and the sequence is playing. In that case the window scrolls automatically.

[see Bar Editor / Scroll]

End Line

Every sequence has an End line. The End line appears as a stationary vertical dotted yellow line and marks the last measure at the end of a sequence. The End line is actually an event and can be dragged like any other event. The length of a sequence is changed by moving the end line.

When the Time line reaches the End line during playback, the sequence stops or repeats. Events crossing the end line are allowed to play for their full duration, rather than being artificially clipped, even while the sequence repeats. Any events starting after the End line are ignored.

Relative sequences always end on a measure boundary. If the End line event is placed in the middle of a measure, then the sequence will finish playing that measure before stopping or repeating.

Absolute sequences may end at any point.

Current Event

In the Bar Editor, the current event is chosen by clicking on it in the score window. The event will change in appearance to show that it's selected, and the Virtual Sliders and Channel Square will configure themselves to the event's parameters.

[see Bar Editor / Virtual Sliders]

[see Bar Editor / Channel Square]

Dragging Events

One of the main ways to edit events in the Bar Editor is by dragging them. Most events can be dragged horizontally and vertically with the mouse. Dragging an event horizontally will change its location in time. Dragging it vertically will change some parameter based on the type of event. (For instance: Notes would be transposed in half steps, while Aftertouch events would be increased or decreased by one pressure unit.)

When you drag a Note event, you'll hear the note play at each new position (unless **Quiet** is on). This is called "feedback", and is provided to help the ear when editing a sequence.

[see Bar Editor / Options Menu / Feedback]

Dragging the current event beyond the left or right edge of the graphic score window causes the window to scroll through time in the direction of the event. Dragging the current event beyond the upper or lower edge of the graphic score window causes the window to scroll by octaves in the direction of the event.

For events with durations (Notes, Play Sequence, etc.), holding the SHIFT key while dragging the event's tail will change the event's duration.

Cursor keys

The cursor keys can be used in the Bar Editor as an alternative to dragging the current event. This can sometimes give you more control when making small precise movements. Once the current event has been chosen, the CursorLeft and CursorRight keys will move the event horizontally through time. The CursorUp and CursorDown keys will move the current event vertically in whatever increments are applicable to that event type.

For events with durations, the key combinations: (SHIFT-CursorLeft), and (SHIFT-CursorRight) will alter the current event's duration as if its tail was being dragged.

For Note events, the key combinations: (SHIFT-CursorUp) and (SHIFT-CursorDown) will transpose the events by octaves.

◆ Note: When **Snap** is on, events will be moved through time, and durations may be changed in increments of the current Grid Size. When **Snap** is off, events can be moved through time and durations can be adjusted by increments of one clock. Depending on the current **Zoom** setting, this may be too slight a movement to see. Watch the **T=** and **D=** indicators below the score window to see exactly what you are doing.

[see Bar Editor / Snap]

[see Bar Editor / Grid]

[see Bar Editor / Zoom+ and Zoom-]

[see Bar Editor / T=]

[see Bar Editor / D=]

Event Types

The event types are like an alphabet from which a sequence is built. Becoming familiar with all the event types, their effect, and how to manipulate them is essential to an effective editing style. This subchapter will lay out the event types in an orderly fashion so that there is no doubt as to what they do. As you read this, please have the software running, and feel free to experiment with each event type as it is explained, by Adding it to the score, and then dragging the sliders etc.

[see Bar Editor / Add]

Every event type has a name and a mnemonic. The mnemonic is a short word, like a nickname, that will be used later when editing in the Event Editor.

[see Event Editor / Event Types]

There are three classes of events in Music-X.

First, Channel Events that send standard MIDI messages as described in the MIDI specification. These are:

<u>Name</u>	<u>mnemonic</u>
Note	(NOTE)
Channel Aftertouch	(CAT)
Polyphonic Aftertouch	(PAT)
Control Change	(CTL)
Program Change	(PGM)
Pitch Bend	(PBEN)

Second, System Exclusive Events that send MIDI messages understood only by the particular instruments they were intended for. System Exclusive message formats

are designed and specified by the instrument manufacturers. All System Exclusive messages fall under the event type:

System Exclusive (SYSX)

Third, Music-X Events that control the sequencer in some way. These are:

End	(END)
Splice	(SPLI)
Set Repeats	(REPT)
Stop	(STOP)
Tempo Change	(TEMPO)
Time Signature Change	(TSIG)
Change Keymap	(KMAP)
Play Sequence	(PSEQ)
Mute Sequence	(MSEQ)
Solo Sequence	(SSEQ)
Mute Track	(MTRK)
Solo Track	(STRK)

The following is a series of paragraphs explaining what each event type does, how it looks in the graphic score window, what parameters it has, and how to edit it in the Bar Editor.

Note

(mnemonic = NOTE)

Note events are Channel events that send MIDI Note On and Note Off messages to MIDI synthesizers. Note events can also be redirected internally to play Amiga samples.

Note events appear in the Bar editor as colored horizontal bars. When selected or grabbed, they appear hollow.

Note events have six parameters: Start Time, MIDI channel, MIDI key number (pitch), Attack Velocity, Release Velocity, and Duration. All of these parameters may be edited in the Bar Editor.

The virtual sliders take on these parameters: MIDI key number, Attack Velocity and Release Velocity. They are titled **NT:** (for “note”), **ATV:**, and **RVL:** respectively.

Start Time: To change the starting time of a single note, drag it to the left or right. The **T=** field will take on the start time of the current event and change as the event is dragged. You can also move the current Note event or a group of selected Note events through time with the **CursorLeft** and **CursorRight** keys. The **Snap** tool, when on, limits the movements of Note events to increments of the current Grid Size. The start times of groups of Note events may also be changed with the **Quantizer** module.

[see Bar Editor / Snap]

[see Bar Editor / Quantizer Module]

MIDI Channel: The channel number of the current Note event is indicated by the Channel Square and by the color of the event. To change the channel of the current Note event or of a group of selected Note events, click on the numbers in the Channel Square. The color of a Note event changes as its MIDI channel number changes.

[see Bar Editor / Channel Square]

◆ **Note:** All the screens in Music-X use eight colors. Two of those colors (light blue and dark blue) are used for the background in the graphic score window. That means there are six colors left over to represent 16 MIDI channels. Higher channels use the same colors as lower channels. When editing a sequence containing events on different channels using the same color, remember the Channel Square always shows the channel number of the current event and that any channel(s) can be temporarily locked out of the display using the **Lock** tool.

[see Bar Editor / Lock]

MIDI Key Number: The MIDI key number of a Note event is represented by its relative height in the score window. The higher the bar on the screen, the higher the MIDI key number and, most likely, the musical pitch (unless your synth is retuned backwards). Possible key numbers range from 000 to 127 (middle C is key number 60). The exact key number is indicated by the virtual slider titled **NT:**. To change the key number of a Note event, drag it higher or lower in the score, or move the virtual slider. You may also change the key numbers of the current Note or a group of selected Notes with the CursorUp and CursorDown keys. Holding SHIFT while using the CursorUp and CursorDown keys will transpose the current event, or group of selected events, by octaves. You may also move the current Note event to a particular key number by playing that key on the mother keyboard. This last method can be used while in play.

Attack Velocity and Release Velocity: Both of these parameters are displayed in the same way and only one may be shown at a time. They are switched using the **Display** menu. Velocities are displayed on the screen as dark blue rectangles extending upwards from the bottom of the score. The higher the rectangle, the faster the attack velocity. Each velocity is displayed directly below or behind its related Note event. Two of the virtual sliders configure to attack and release velocities and are titled **AVL:**, and **RVL:**. The attack and release velocities of the current Note event may be changed by moving these sliders. The velocities of a group of Notes may be changed with the **Velocity Scaling** module. Attack Velocities can range from 001 to 127. True Release Velocities can range from 000 to 127. The slider also allows a release velocity of 128 to mark a special case (see note below).

[see Bar Editor / Display Menu / Attack Velocity]

[see Bar Editor / Display Menu / Release Velocity]

[see Bar Editor / Velocity Scaling Module]

Duration: A single Music-X Note event actually sends two MIDI messages: a Note On message and a Note Off message. The duration of the Note event is the length of time between the sending of these two messages. The duration of Note events is represented by their size; the longer the bar, the longer the note. The duration of the current Note event (when grabbed) is also shown in the **D=** field underneath the score window. Change duration by holding the SHIFT key while dragging the note tail. The durations of the current Note event or of a group of selected Note events may be changed by holding the SHIFT key and using the CursorLeft and CursorRight keys. The **Snap** tool limits the changing of durations of Note events to increments of the current Duration, as set in the Grid requester. The Durations of Note events may also be changed with the Quantizer Module.

♦ **Note:** Many synthesizers send Note Off messages as Note On messages with attack velocities of zero. This is a standard technique that makes the MIDI data stream more efficient by eliminating the need to change the “running status”. Music-X records Note Offs in whichever form it receives them. A Music-X Note event will show a Release Velocity of 128 if the Note Off was received as a Note On with zero velocity. This value of 128 would be illegal in MIDI, and is only used as a “flag” by Music-X so it knows to send the Note Off as a Note On during playback. Any other release velocity would cause the Note Off to be sent normally. This parameter is fully editable the same as any other; a Note Off recorded in one form may be changed and played back in the other form.

Channel Aftertouch

(mnemonic = CAT)

Channel Aftertouch events send Channel Aftertouch messages to MIDI instruments.

Channel Aftertouch events appear as colored dots connected by vertical lines to the bottom of the score window. When grabbed or selected, the dots appear hollow.

Channel Aftertouch events have three parameters: Start Time, MIDI Channel, and Pressure.

One virtual slider takes on the Pressure parameter and is titled **PRS:**. The other sliders are ignored.

Start Time: To change the start time of a Channel Aftertouch event, grab its head (the dot) and drag it horizontally. After a Channel Aftertouch event has been grabbed, its start time will be indicated in the **T=** field below the score window. The start time of the current Channel Aftertouch event or of a group of selected Channel Aftertouch events may also be changed with the **CursorLeft** and **CursorRight** keys. (The movements of Channel Aftertouch events are not affected by **Snap**.)

MIDI Channel: The channel number of the current Channel Aftertouch event is indicated by the Channel Square and by the color of the event. To change the MIDI channel number of the current Channel Aftertouch event or of a group of selected Channel Aftertouch events, click on any number in the Channel Square. The color of a Channel Aftertouch event will change as its MIDI channel number changes.

Pressure: This parameter relates to how hard the key is being pressed after reaching the bottom of travel. It can range from 000 to 127. This value is indicated by the relative height of the Channel Aftertouch event in the score window and by the virtual slider. To change the Pressure value of a Channel Aftertouch event, drag the event vertically, or move the virtual slider titled **PRS:**, or press the **CursorUp** and **CursorDown** keys. The pressure parameter of Channel Aftertouch events may also be manipulated using the Aftertouch Scaling Module.

[see Bar Editor / Aftertouch Scaling Module]

Polyphonic Aftertouch

(mnemonic = PAT)

Polyphonic Aftertouch events send Polyphonic Aftertouch messages to MIDI instruments.

Polyphonic Aftertouch events are like Channel Aftertouch events except that differ-

ent keys may have different pressure values. Therefore they have an extra parameter for MIDI key number.

Polyphonic Aftertouch events look the same as Channel Aftertouch events; colored dots connected by vertical lines to the bottom of the score window that become hollow when grabbed or selected. Only one form of Aftertouch event (Channel or Polyphonic) may be shown in the graphic score window at a time. The **Display** menu is used to control which event type is to be shown.

Polyphonic Aftertouch events have four parameters: Start Time, MIDI Channel, MIDI key number, and Pressure.

Two virtual sliders take on the parameters of MIDI key number, and Pressure. They are titled **NT:**, and **PRS:**, respectively. The last slider is ignored.

Start Time: The start time of the current Polyphonic Aftertouch event is indicated in the **T=** field. To change the start time of a Polyphonic Aftertouch event, grab its head and drag it horizontally. The start time of the current Polyphonic Aftertouch event or of a group of selected Polyphonic Aftertouch events may also be changed with the **CursorLeft** and **CursorRight** keys. (The movements of Polyphonic Aftertouch events are not affected by **Snap**.)

MIDI Channel: As with all channel events, the channel number of the current Polyphonic Aftertouch event is indicated both by the Channel Square, and by the color of the event. To change the MIDI channel number of the current Polyphonic Aftertouch event or of a group of selected Polyphonic Aftertouch events, click on the numbers in the Channel Square. The color of a Polyphonic Aftertouch event will change as its MIDI channel number changes.

MIDI key number: The MIDI key number of the current Polyphonic Aftertouch event is indicated by the virtual slider titled **NT:**. To change the MIDI key number, move the slider. Possible key numbers range from 000 to 127.

Pressure: The pressure value is indicated by the relative height of the Polyphonic Aftertouch event in the score window, and by the virtual slider. Possible pressure values range from 000 to 127. To change the pressure value of a Polyphonic Aftertouch event, drag the event vertically or move the virtual slider titled **PRS:**. You may also change the pressure value with the CursorUp and CursorDown keys. The pressure values of a group of Polyphonic Aftertouch events may be manipulated with the Aftertouch Scaling Module.

[see Bar Editor / Aftertouch Scaling Module]

Control Change

(mnemonic = CTL)

Control Change events send Control Change messages to MIDI instruments. A Control Change message causes the sound of an instrument to change as if one of the controllers were moved on the instrument itself. Controllers are things like the modulation wheel, breath controller, sustain pedal, etc. Each controller is identified by a number. This family of controllers does not include the pitch bend wheel, which has its own messages, nor does it include the front panel switches used to modify synthesizer parameters making up a program (VCF cutoff, envelope levels, etc.). Synthesizer parameters can be controlled via System Exclusive messages.

Control Change events appear as colored dots connected by vertical lines to the bottom of the score window, and become hollow when grabbed or selected. Control Change events are displayed using the same graphic icons as Channel Aftertouch and Polyphonic Aftertouch events. To avoid confusion, only one of these three event types may be displayed and edited at a time. The Display menu is used to control which type is shown.

[see Bar Editor / Display menu]

Control Change events have four parameters: Start Time, MIDI Channel, Controller Number, and Value.

Two virtual sliders take on the parameters of Controller Number, and Value. They are titled **CTL:**, and **VAL:** respectively. The last slider is ignored.

Start Time: The start time of a Control Change event can be changed by dragging the event horizontally. The CursorLeft and CursorRight keys will move the current event through time as well. The start time of the current Control Change event is indicated in the **T=** field below the score window, and changes as the event is dragged. (The movements of Control Change events are not affected by **Snap**.)

Controller Number: Each controller is identified by a number. This number specifies the particular controller you wish to change with this message. The controller number for the current Control Change event is indicated by the virtual slider titled **CTL:**. To change this parameter, move the slider. The range of the slider is from 000 to 127.

Value: The Value parameter represents different things depending on the controller number, but is basically an intensity indicator. The value is indicated by the relative height of the Control Change event in the score window and by the virtual slider. To change the value of the current Control Change event, drag the event vertically or move the virtual slider titled **VAL:**. You may also change the value with the CursorUp and CursorDown keys. The range of possible values is from 000 to 127.

Normal controller numbers range from 0 to 120. Controllers 121 to 127 are “Mode Messages”. Here is a partial list of some of the more common controllers.

<u>Controller</u>	<u>Name</u>	<u>Value</u>
1	Modulation Wheel	000-127
2	Breath Controller	000-127
4	Foot Controller	000-127
7	Volume	000-127
8	Balance	left=0, equal=64, right=127
10	Pan	left=0, center=64, right=127
11	Expression Controller	000-127

64	Hold (Sustain) Pedal	Off=0 On=127
65	Portamento Pedal	Off=0 On=127
66	Sostenuto Pedal	Off=0 On=127
96	Data Increment Slider 1	000-127
97	Data Increment Slider 2	000-127
121	Reset All Controllers	000
123	All Notes Off	000

◆ Note: A complete list is available from the International MIDI Association. Consult the manufacturer's documentation to see if your particular MIDI instrument sends or responds to particular control changes.

◆ Note: Some of these controllers, and others, can be sent directly from the Filters Page using the MIDI Message Panel.

[see Filters Page / MIDI Message Panel]

Program Change

(mnemonic = PGM)

Program Change events send Program Change messages to MIDI instruments. A Program Change message causes a MIDI instrument to recall one of its memorized sound settings, thus changing its timbre.

Program Change events appear as colored text written in the score window. The abbreviation "PGM:" is followed by the program number. Example: **PGM: 091**. When a Program Change event is grabbed or selected, it appears written in negative. Program Change events can be grabbed by clicking on the "PGM:" portion, rather than on the program Number.

Program Change events have three parameters: Start Time, MIDI Channel, and Program Number.

One virtual slider takes on the Program Number parameter, and is titled **PGM:**. The other sliders are ignored.

Start Time: As with all events, dragging a Program Change event horizontally will change its start time. For the current event, pressing the CursorLeft and CursorRight keys will also move it through time. The start time of the current event is indicated in the **T=** field below the score window, and changes as the event is dragged or moved. (The movements of Program Change events are not affected by **Snap**.)

MIDI Channel Number: The MIDI channel number of the current Program Change event is indicated by the color of the event and by the Channel Square. To change the channel of the current event, click on the numbers in the Channel Square; the event's color will change accordingly.

Program Number: The program number is written in the Program Change event itself. When a Program Change event is grabbed, and becomes the current event, its program number is also indicated by the virtual slider titled **PGM:**. To change the program number, drag the event vertically or move the slider, or press the CursorUp and CursorDown keys. The range of program numbers that can be sent is from 001 to 128.

◆ **Note:** Some synthesizers number their programs using a system called “octal plus one”, or simply “octal”. What this means is that they use combinations of the digits 1 through 8 to come up with 64 unique program numbers (11 through 18, then 21 to 28, and so on until 81-88). These synthesizers still respond to Program Change messages in a logical way. Their lowest program number corresponds to program change number 001 and increases sequentially. For example, octal 11-18 = Program 001-008, octal 21-28 = Program 009-016, etc.

Pitch Bend

(mnemonic = PBEN)

Pitch Bend events send MIDI pitch bend messages to MIDI instruments. Pitch bend is used to temporarily raise or lower the pitch of an instrument while it plays. This is used to get the “blue notes” found between the normal pitches of the keys on a piano.

Pitch Bend events appear as small diamonds, colored according to their MIDI channel. When grabbed, they become hollow.

Pitch Bend events have three parameters: Start Time, MIDI Channel, and Bend Value.

The first virtual slider takes on the parameter of Bend Value, and is titled **BND:**. The other sliders are ignored.

Start Time: Drag a Pitch Bend event horizontally to change its start time. The start times of a single Pitch Bend event or a group of selected Pitch Bend events may be changed by pressing the CursorLeft and CursorRight keys. The start time of the current event is indicated in the **T=** field below the score window, and changes as the event is dragged or moved. (The movements of Pitch Bend events are not affected by **Snap**.)

MIDI Channel: The MIDI Channel of the current Pitch Bend event is indicated by the event’s color and by the Channel Square. Click on the Channel Square to change the current event’s MIDI Channel.

Bend Value: This parameter ranges from -8192 (maximum lowering of pitch), to 0000 (normal pitch), to +8191 (maximum raising of pitch). Bend Value is indicated by the relative height of the event in the score window. The bend value for the cur-

rent Pitch Bend event is also indicated by the virtual slider titled **BND**:. To change the Bend Value, drag the event vertically, or move the virtual slider, or press the CursorUp and CursorDown keys.

◆ **Note:** Most synthesizers have a pitch bend sensitivity parameter that governs how they will respond to the maximum values in a Pitch Bend message. The setting of this parameter would change the effect of a Pitch Bend event.

System Exclusive

(mnemonic = SYSX)

System Exclusive events send MIDI messages understood only by the particular instruments they were intended for. They can have various effects that depend entirely on the specifications of the instrument manufacturers.

System Exclusive events are more properly edited in the Event Editor.

In the Bar Editor, System Exclusive events appear as a vertical dash spanning about two pitch lines.

The only parameter of System Exclusive events adjustable in the Bar Editor is Start Time. System Exclusive events can be moved by first selecting them with the **Select** tool, and then pressing the CursorLeft and CursorRight keys. System Exclusive events cannot be dragged. The **Snap** tool, when on, limits the movements of System Exclusive events to increments of the current Grid Size.

[see Event Editor / Event Types / SYSX]

End

(mnemonic = END)

Every sequence has an End event. The End event is used to mark the end of musical time in each sequence. When found during playback, the End event causes Music-X

to either repeat or stop the sequence. The End event may be placed anywhere in the sequence but behaves differently depending on the type of sequence it's in. Relative sequences will always stop or repeat on measure boundaries. Absolute sequences will either stop or repeat as soon as the End event is found. Events starting after the End event are ignored during playback.

The End line appears as a vertical yellow dotted line. While being dragged, the line turns green, and a temporary red line marks the original position. End events are always placed in the same vertical position and cannot be dragged vertically.

The End event has only one parameter: Start Time.

The virtual sliders are ignored.

Start Time: The start time of the End event (the point at which it occurs in the sequence) is changed by dragging the event horizontally, or, while the End event is the current event, by pressing the CursorLeft and CursorRight keys. The start time of the End event is indicated in the T= field below the score window, and changes as the event is dragged or moved. The **Snap** tool, when on, limits the movement of the End event to increments of the current Grid Size.

Splice

(mnemonic = SPLI)

Splice events are harmless do-nothing events, imbedded in the sequence by Music-X under certain conditions while recording, as a marker to aid the user when editing the sequence later.

Splice events appear as stationary vertical green lines. When grabbed or dragged, they also appear green.

Splice events have two parameters: Start Time, and a special message parameter read only in the Event Editor.

The virtual sliders are ignored.

Start Time: The start times of the Splice events can be changed by dragging the event horizontally, or, while a Splice event is the current event (after having been grabbed and released), by pressing the CursorLeft and CursorRight keys. The start time of the Splice event is indicated in the **T=** field below the score window, and changes as the event is dragged or moved. The **Snap** tool, when on, limits the movements of Splice events to increments of the current Grid Size.

Splice events are entered into a sequence when recording from the Sequencer Page in three cases: When recording in Manual Punch-In Mode, when recording in Auto Punch-In Mode, and when recording in Loop Mode.

When recording in one of the Punch-In modes, Music-X enters a Splice event into the sequence when the actual “punch” begins, and when it ends. When recording in Loop mode, Music-X enters a Splice event into the sequence at the beginning of each loop.

When editing the sequence later in the editors, the Splice events can be used as reference markers to see where the punch points or loops occurred. This can come in handy when cutting and pasting.

Set Repeats

(mnemonic = REPT)

The Set Repeats event tells Music-X how many times a sequence should be repeated during playback. If there is no Set Repeats event in the sequence then the sequence will play once and stop.

The Set Repeats event appears as text written in the score window. The abbreviation “REP:” is followed by a repeat number. Example: **REP: 16**. Normally the Set Repeats event appears as white text with a black shadow. When the event is grabbed

or selected, the shadow turns yellow. Set Repeats events are always placed in the same vertical position and cannot be dragged vertically.

The Set Repeats event has two parameters: Start Time, and Repeats.

One virtual slider takes on the Repeats parameter, and is titled **RPT:**. The other sliders are ignored.

Start Time: The Start Time of a Set Repeats event is not really important. As long as it happens sometime before the End event, the sequencer will be able to use it correctly. It might be convenient to keep Set Repeats events at the beginnings of sequences so you can find them quickly. To change the start time of a Set Repeats event, drag the event horizontally. You may also move the current event or group of selected events through time with the CursorLeft and CursorRight keys. The **T=** field will indicate the start time of the current event. The **Snap** tool, when on, limits the movements of Set Repeat events to increments of the current Grid Size.

Repeats: This parameter ranges from 1 to 100. Values 1 through 99 represent the total number of times the sequence will be played. A value of 100 causes the sequence to repeat indefinitely. The value of this parameter is indicated in the event itself. When the current event is a Set Repeats event, the repeats value is also indicated by the virtual slider titled **RPT:**. To change the Repeats parameter, move the slider.

♦ Note: If more than one Set Repeats event is added to a sequence, the last event (preceeding the End line) takes precedence.

Stop

(mnemonic = STOP)

When a Stop event is encountered in any playing sequence, the sequencer will react as if the **STOP** button was clicked. All sequences will stop playing and the clock will stop running. This can be used at the end of a song.

Stop events cannot be added from the Bar Editor (they must be entered from the Event Editor). However, they may be seen and edited in the Bar Editor.

The Stop event appears in the Bar Editor as text written in the score window. The word **STOP** is written in white with a black shadow. When grabbed, or selected, the event's shadow turns yellow. Stop events always appear in the same vertical position and cannot be dragged vertically.

This event has one parameter: Start Time.

The virtual sliders are ignored.

Start Time: This parameter determines the point in the sequence at which the Stop event will occur. To change the start time of a Stop event, drag the event horizontally. The **T=** field will show the current start time as the event is moved. The current Stop event or a selected Stop event may be moved through time with the CursorLeft and CursorRight keys. The **Snap** tool, when on, limits the movement of a Stop event to increments of the current Grid Size.

Tempo Change

(mnemonic = TEMPO)

Tempo Change events are Music-X events that cause the sequencer to change the Relative clock rate and therefore the tempo of Relative sequences. Absolute sequences are not affected by Tempo Changes. Tempo Change can be used to set a preferred tempo at the beginning of a piece, or to create accelerando and decelerando effects during the body of the piece.

Tempo Change events look similar to Note events except that they are thicker (about two pitch lines high), and are bright green with white lettering. When a Tempo Change event is grabbed or selected, it appears black with a green border.

A Tempo Change event has three parameters: Start Time, New Tempo, and Duration.

One virtual slider takes on the New Tempo parameter. The other virtual sliders are ignored.

Start Time: This parameter determines when in the sequence the tempo will start to change. To move the start time of a Tempo Change event, drag the event horizontally. The **T=** field will show the current start time as the event is moved. The current Tempo Change event or one or more selected Tempo Change events may also be moved through time with the **CursorLeft** and **CursorRight** keys. The **Snap** tool, when on, limits the movements of Tempo Change events to increments of the current Grid Size.

New Tempo: This parameter ranges from 10 to 300 quarter notes per minute. The current value of this parameter is indicated in three ways. First, by the relative height of the event in the score window. Second, the new tempo number is written on the event itself when it is large enough, followed by an equals sign (“=”) and the words “NEW TEMPO”. Third, when a Change Tempo event is grabbed and dragged, the virtual slider titled **TMP:** shows the new tempo value. To change this parameter, drag the Tempo Change event vertically, move the virtual slider, or press the **CursorUp** and **CursorDown** keys.

Duration: When a Tempo Change event has no duration (duration=0), the new tempo is achieved instantly. This effect is sometimes called “metric modulation”, and is found profusely in progressive rock and 20th century orchestral works. The more subtle effects of “accelerando” and “decelerando” are achieved by giving the Tempo Change event a duration. The way this works is: The tempo setting is automatically increased or decreased at a regular rate throughout the duration of the event, starting at whatever the current tempo happens to be, and arriving at the new tempo exactly at the end of the event. The rate of change from the current tempo to

the new tempo is based on the difference between the two tempos and the duration of the Tempo Change event. A Tempo Change event of the same duration would change at a faster rate if going to a more distant tempo than if going to a nearer one. The durations of Tempo Change events are changed the same way Note event durations are changed.

◆ Note: Only one Tempo Change event is played at a time by Music-X. A new Tempo Change event takes precedence over a currently playing Tempo Change event. It's possible to interrupt one event with another by overlapping their durations, without worrying about conflicting ramps.

◆ Note for math buffs: The human mind perceives tempo changes exponentially. That is, to create the same perceived effect, a tempo must be increased by multiplication rather than by addition. In Music-X the tempo is calculated by addition, but the rate at which that addition is made speeds up as the sequence does. This creates an exponential curve. That means the perceived rate of change is equal throughout the duration of the Tempo Change event. It's possible to get a purely linear increase by playing a Tempo Change event from an Absolute sequence; the Tempo Change event will not speed up as everything else does.

Time Signature Change

(mnemonic = TSIG)

Time Signature Change events cause Music-X to change the global time signature that it uses to interpret all sequences during playback.

◆ Note: Future versions of Music-X may allow individual sequences to have their own independent time signatures. For reasons of upward compatibility, it would be a good idea to include Time Signature Change events in all your sequences when Time Signature Change events are used, though they are currently redundant.

Time Signature Change events appear as text written in the score window. The abbreviation "SIG:" is followed by two numbers representing the new time signa-

ture. Example: **SIG: 5/4**. Time Signature Change events are normally written in white, with a black shadow. When a Time Signature Change event is grabbed or selected, the shadow turns yellow. Time Signature Change events can be grabbed by clicking on the “SIG:” portion, rather than on the number portion. Time Signature Change events are always placed in the same vertical position and cannot be dragged vertically.

Time Signature Change events have three parameters: Start Time, Numerator, and Denominator.

Two of the virtual sliders take on the parameters of Numerator, and Denominator. These are titled **NUM:**, and **DEN:**, respectively. The last slider is ignored.

Start Time: To change the start time of a Time Signature Change event, drag it horizontally in the score window, or, while it is the current event, press the CursorLeft and CursorRight keys. The start time of a Time Signature Change event is indicated in the **T=** field below the score window when the event is grabbed, and changes as the event is dragged or moved. The **Snap** tool, when on, limits the movements of Time Signature Change events to increments of the current Grid Size.

Numerator: This parameter represents the top number in the time signature, showing the number of beats in a measure. this number can range from 1 to 64 but can be no greater than four times the denominator. The numerator is indicated in the Time Signature Change event itself, and, when a Time Signature Change event is grabbed, by the virtual slider titled **NUM:**. To change the numerator, move the virtual slider.

Denominator: This parameter represents the bottom number in the time signature, showing the type of note value used to make one beat. This number is indicated in the Time Signature Change event itself and, when a Time Signature Change event is grabbed, by the virtual slider titled **DEN:**. The slider ranges from 0 to 4. Think of this number as the power of two needed to express a musical note value. (0=whole note, 1=half note, 2=quarter note, 3=eighth note, 4=sixteenth note.) To change the denominator, move the virtual slider.

Change Keymap

(mnemonic = KMAP)

Change Keymap events cause Music-X to reroute new information coming from the mother keyboard (or any MIDI input device) through one of the four keymaps in the Filters Page. This can be used for playing live along with a prerecorded song. By playing back a sequence containing Change Keymap events, the whole configuration of the mother keyboard can be changed automatically at different song sections.

Example: Use the Keymap Editor to set up two keymaps. One keymap that allows you to play bass with the left hand and piano chords with the right, and another keymap that allows you to play bass and a solo sound instead. Then put Change Keymap events at the appropriate points in your song to change keymaps automatically on playback.

[see Filters Page / Keyboard Map]

[see Keymap Editor chapter]

Change Keymap events appear as text written in the score window. The word “MAP:” is followed by a keymap number. Example: **MAP: 1**. Change Keymap events are normally written in white, with a black shadow. When a Change Keymap event is grabbed or selected, the shadow turns yellow. Change Keymap events can be grabbed by clicking on the “MAP:” portion, rather than on the map number. Change Keymap events are always placed in the same vertical position and cannot be dragged vertically.

Change Keymap events have three parameters: Start Time, MIDI Channel, and Keymap Number.

Two of the virtual sliders take on the parameters of MIDI Channel, and Keymap number. These are titled **CHN:**, and **MAP:**, respectively. The last slider is ignored.

Start Time: To change the start time of a Change Keymap event, drag it horizontally in the score window, or, while it is the current event, press the CursorLeft and CursorRight keys. The start time of the current Change Keymap event is indicated in the **T=** field below the score window, and changes as the event is dragged or moved. The **Snap** tool, when on, limits the movements of Change Keymap events to increments of the current Grid Size.

MIDI Channel: This parameter determines which incoming MIDI channel will now be routed through a keymap. This would usually be set to the channel that the mother keyboard is transmitting on. If there are multiple input devices then multiple Change Keymap events can be used. The MIDI Channel parameter of the current Change Keymap event is indicated by, and edited with, the virtual slider titled **CHN:**.

Keymap number: This parameter determines which of the four keymaps will be used to process incoming data on the specified channel. There are five choices: the four keymaps (1-4) and a setting of zero (0) which means “no keymap”. When a Change Keymap event specifying a Map of zero is played in a sequence, keymaps for the MIDI channel specified in the event are turned off, and data is routed through the Filters Page normally. The Keymap number parameter is indicated by, and edited with, the virtual slider titled **MAP:**.

Play Sequence

(mnemonic = PSEQ)

Play Sequence events cause Music-X to initiate the playing of new sequences, allowing one sequence to “spawn” others. The duration and transposition of each new sequence is specified in the Play Sequence event, and, multiple Play Sequence events may be played simultaneously. This allows you to think of, and treat, whole sequences as if they were simple Note events. By stringing Play Sequence events one after the other, “Control” sequences can be created.

[see Advanced Users / Song Assembly]

Play Sequence events look much like Note events (like colored horizontal bars), except that they are slightly thicker, and they're magenta with white lettering. When grabbed, or selected, Play Sequence events are black with magenta borders.

Play Sequence events have five parameters: Start Time, Sequence, Transposition, Cut, and Duration.

Two of the virtual sliders take on the parameters Sequence Number and Transposition. They are titled **SEQ:**, and **KEY:** respectively.

Start Time: To change the start time of a Play Sequence event, drag it horizontally, or, for the current Play Sequence event or a group of selected Play Sequence events, press the CursorLeft and CursorRight keys. The start time of the current Play Sequence event (when grabbed) will also be indicated by the **T=** field below the score window. The **Snap** tool, when on, limits the movements of Play Sequence events to increments of the current Grid Size. The Quantizer Module can also be used to alter the start times of Play Sequence events.

◆ Note: By spawning multiple versions of the same sequence, at slightly different times, one can create a MIDI delay effect.

Sequence: This parameter determines which of the 250 sequences in memory will be spawned by the current Play Sequence event. This parameter is indicated in a few ways. First, the sequence number and name are written on the event itself, in white lettering. Second, the virtual slider titled **SEQ:** shows the sequence number for the current event. Third, the event's vertical position in the score window gives some indication as to which sequence will be played. There are not enough vertical screen positions to represent all 250 possible sequences. Therefore, Play Sequence events have been limited to the first 16 vertical positions. To change the sequence number of the current event, move the virtual slider or press the CursorUp and CursorDown keys.

◆ Note: If an empty sequence is specified in the sequence parameter, nothing will be heard.

Transposition: This parameter is used to play the spawned sequence in other than its original key. Transposition is indicated as a halfstep offset from the original key. The range is from -64 (64 halfsteps lower), to 000 (no transposition), to +63 (63 halfsteps higher). Transposition is shown in two ways. First, the transposition number is written on the event itself, after the sequence number and name, and enclosed in brackets. Second, the virtual slider titled **KEY:** indicates the halfstep offset for the current Play Sequence event. To change the transposition parameter, drag the virtual slider.

◆ Note: MIDI specifies 128 different possible note numbers (000-127). Notes, played by a spawned sequence, that have been transposed beyond the limits of the MIDI specification “wrap around” to stay legal. Example: MIDI key 127, after being transposed up a halfstep, would be played as MIDI key 000.

Duration: The duration of the Play Sequence event determines the length of time that the spawned sequence will play before being stopped. If the duration of the Play Sequence event is longer than the number of measures in the sequence to be spawned, the spawned sequence will automatically repeat. (Relative sequences always repeat on measure boundaries; Absolute sequences repeat from the End line.) Duration is indicated by the length of the event; the longer the event’s tail, the longer its duration. The duration of the current Play Sequence event (when grabbed) is also indicated by the **D=** field underneath the score window. To change the duration of the current Play Sequence event, hold the SHIFT key while dragging the event’s tail. You may also change the durations of the current Play Sequence event or of a group of selected Play Sequence events by holding the SHIFT key and using the CursorLeft and CursorRight keys. The **Snap** tool limits the changing of durations of Play Sequence events to increments of the current Duration, as set in the Grid requester. The Quantizer module may also be used to alter the durations of Play Sequence events.

◆ Note: As with all event types that have durations, Play Sequence events starting before the End line and continuing after the End line are played for their full duration rather than being cut off artificially when the sequence stops or repeats.

Cut: This parameter is used to determine how the spawned sequence will cut off (stop playing) when the Play Sequence event ends. The Cut parameter cannot be edited from the Bar Editor; use the Event Editor instead.

◆ **Note:** Play Sequence events are “self referencing” events, in that a sequence containing Play Sequence events can spawn other sequences also containing Play Sequence events, etc. This is discussed in more detail in the Event Editor chapter and in the Advanced Users chapters.

[see Event Editor / Event Types / PSEQ]

[see Advanced Users / Song Assembly]

Mute Sequence

(mnemonic = MSEQ)

A Mute Sequence event causes Music-X to temporarily mute any and all tracks that are currently playing a specified sequence during play or record modes, whether that sequence is self-starting or has been spawned by another sequence.

◆ **Note:** The Track Window in the Sequencer Page will show these tracks as muted by displaying them in green.

Mute Sequence events look like Play Sequence events except that they are yellow. When grabbed, or selected, Mute Sequence events are black with yellow borders.

Mute Sequence events have three parameters: Start Time, Sequence, and Duration.

One of the virtual sliders takes on the Sequence parameter, and is titled **SEQ:**. The other sliders are ignored.

Start Time: To change the start time of a Mute Sequence event, drag it horizontally, or, for the current Mute Sequence event or group of selected Mute Sequence events, press the CursorLeft and CursorRight keys. The start time of the current Mute Se-

quence event (when grabbed) will also be indicated by the **T=** field below the score window. The **Snap** tool, when on, limits the movements of Mute Sequence events to increments of the current Grid Size. The Quantizer Module can also be used to alter the start times of Mute Sequence events.

Sequence: This parameter determines which of the 250 sequences in memory will be muted by the current Mute Sequence event. This parameter is indicated in a few ways. First, the sequence number and name are written on the event itself, in red lettering. Second, the virtual slider titled **SEQ:** shows the sequence number for the current event. Third, the event's vertical position in the score window gives some indication as to which sequence will be played. There are not enough vertical screen positions to represent all 250 possible sequences. Therefore, Mute Sequence events have been limited to the first 16 vertical positions. To change the sequence number of the current event, move the virtual slider or press the CursorUp and CursorDown keys.

Duration: The duration of the Mute Sequence event determines the length of time that the sequence will stay muted. As with Note events, the duration of a Mute Sequence event is indicated by the length of its tail. The duration of the current Mute Sequence event (when grabbed) is also indicated by the **D=** field underneath the score window. To change the duration of a Mute Sequence event, hold the SHIFT key while dragging its tail. You may also change the duration of the current Mute Sequence event or of a group of selected Mute Sequence events by holding the SHIFT key and using the CursorLeft and CursorRight keys. The **Snap** tool limits the changing of durations of Mute Sequence events to increments of the current Duration, as set in the Grid requester. The Quantizer module may also be used to alter the durations of Mute Sequence events.

Solo Sequence

(mnemonic = SSEQ)

A Solo Sequence event causes Music-X to temporarily mute all tracks except those playing a specified sequence during play or record modes. (Multiple tracks can be

playing copies of the same sequence if triggered by Play Sequence events.) That means that only those tracks playing the specified sequence will be heard for the duration of the Solo Sequence event. Only one Solo Sequence event can be played by Music-X at a time. In the event that Solo Sequence events overlap, the first one played always takes precedence.

◆ **Note:** One easy way to implement the typical “drum breakdown” section in a dance piece is to use a Solo Sequence event to temporarily solo the drum sequence.

Solo Sequence events look like Mute Sequence events except that they are red. When grabbed, or selected, Solo Sequence events look black with red borders.

Solo Sequence events have three parameters: Start Time, Sequence, and Duration.

One of the virtual sliders takes on the Sequence parameter, and is titled **SEQ:**. The other sliders are ignored.

Start Time: To change the start time of a Solo Sequence event, drag it horizontally, or, for the current Solo Sequence event or group of selected Solo Sequence events, press the CursorLeft and CursorRight keys. The start time of the current Solo Sequence event (when grabbed) will also be indicated by the T= field below the score window. The **Snap** tool, when on, limits the movements of Solo Sequence events to increments of the current Grid Size. The Quantizer Module can also be used to alter the start times of Solo Sequence events.

Sequence: This parameter determines which of the 250 sequences in memory will be soloed by the current Solo Sequence event. This parameter is indicated in a few ways. First, the sequence number and name are written on the event itself, in white lettering. Second, the virtual slider titled **SEQ:** shows the sequence number for the current event. Third, the event’s vertical position in the score window gives some indication as to which sequence will be played. There are not enough vertical screen positions to represent all 250 possible sequences. Therefore, Solo Sequence events

have been limited to the first 16 vertical positions. To change the sequence number of the current event, move the virtual slider or press the CursorUp key or the Cursor-Down key.

Duration: The duration of the Solo Sequence event determines the length of time that the sequence will stay soloed. As with Note events, the Duration of a Solo Sequence event is indicated by the length of its tail. The duration of the current Solo Sequence event (when grabbed) is also indicated by the **D=** field underneath the score window. To change the duration of a Solo Sequence event, hold the SHIFT key while dragging its tail. You may also change the durations of the current Solo Sequence event or of a group of selected Solo Sequence events by holding the SHIFT key and using the CursorLeft and CursorRight keys. The **Snap** tool limits the changing of durations of Solo Sequence events to increments of the current Duration, as set in the Grid requester. The Quantizer module may also be used to alter the durations of Solo Sequence events.

Mute Track

(mnemonic = MTRK)

As explained in the Sequencer Page chapter, a track is something that plays a sequence. Music-X has 20 tracks. As a sequence is initiated during play or record mode, that sequence is assigned to the next available track and is played by that track. A Mute Track event causes Music-X to temporarily mute a particular track, and consequently the sequence being played by that track, during play or record modes.

Like Mute Sequence events, Mute Track events appear as yellow horizontal bars (yellow represents “mute”). When grabbed, or selected, Mute Track events appear black with yellow borders.

Mute Track events have three parameters: Start Time, Track, and Duration.

One of the virtual sliders takes on the Track parameter, and is titled **TRK:**. The other sliders are ignored.

Start Time: To change the start time of a Mute Track event, drag it horizontally, or, for the current Mute Track event or group of selected Mute Track events, press the CursorLeft and CursorRight keys. The start time of the current Mute Track event (when grabbed) will also be indicated by the **T=** field below the score window. The **Snap** tool, when on, limits the movements of Mute Track events to increments of the current Grid Size. The Quantizer Module can also be used to alter the start times of Mute Track events.

Track: This parameter determines which of the 20 tracks in Music-X will be muted by the current Mute Track event. This parameter is indicated in a few ways. First, the track number is written on the event itself, in red lettering. The text “**TRACK = #**” is followed by the track number. For example, “**TRACK = #6**”. Second, the virtual slider titled **TRK:** shows the track number for the current event. Third, by the event’s vertical position in the score window. Mute Track events have 20 vertical positions, one for each track. To change the sequence number of the current event, drag the event vertically, move the virtual slider, or press the CursorUp and CursorDown keys.

Duration: The duration of the Mute Track event determines the length of time that the track will stay muted. As with Note events, the Duration of a Mute Track event is indicated by the length of its tail. The duration of the current Mute Track event (when grabbed) is also indicated by the **D=** field underneath the score window. To change the duration of a Mute Track event, hold the SHIFT key while dragging its tail. You may also change the durations of the current Mute Track event or of a group of selected Mute Track events by holding the SHIFT key and using the CursorLeft and CursorRight keys. The **Snap** tool limits the changing of durations of Mute Track events to increments of the current Duration, as set in the Grid requester. The Quantizer module may also be used to alter the durations of Mute Track events.

Solo Track

(mnemonic = STRK)

A Solo Track event causes Music-X to temporarily mute all 20 tracks except one, during play or record modes.

Like Solo Sequence events, Solo Track events appear as red horizontal bars (red represents “solo”). When grabbed, or selected, Solo Track events appear black with red borders.

Solo Track events have three parameters: Start Time, Track, and Duration.

One of the virtual sliders takes on the Track parameter, and is titled **TRK:**. The other sliders are ignored.

Start Time: To change the start time of a Solo Track event, drag it horizontally, or, for the current Solo Track event or group of selected Solo Track events, press the CursorLeft and CursorRight keys. The start time of the current Solo Track event (when grabbed) will also be indicated by the T= field below the score window. The **Snap** tool, when on, limits the movements of Solo Track events to increments of the current Grid Size. The Quantizer Module can also be used to alter the start times of Solo Track events.

Track: This parameter determines which of the 20 tracks in Music-X will be soloed by the current Solo Track event. This parameter is indicated in a few ways. First, the track number is written on the event itself, in white lettering. The text “TRACK = #” is followed by the track number. For example, “TRACK = #7”. Second, the virtual slider titled **TRK:** shows the track number for the current event. Third, the event’s vertical position in the score window indicates which track is to be soloed. Solo Track events have 20 vertical positions, one for each track. To change the sequence number of the current event, drag the event vertically, move the virtual slider, or press the CursorUp and CursorDown keys.

Duration: The duration of the Solo Track event determines the length of time that the track will stay soloed. As with Note events, the Duration of a Solo Track event is indicated by the length of its tail. The duration of the current Solo Track event (when grabbed) is also indicated by the **D=** field underneath the score window. To change the duration of a Solo Track event, hold the SHIFT key while dragging its tail. You may also change the durations of the current Solo Track event or of a group of selected Solo Track events by holding the SHIFT key and using the CursorLeft and CursorRight keys. The **Snap** tool limits the changing of durations of Solo Track events to increments of the current Duration, as set in the Grid requester. The Quantizer module may also be used to alter the durations of Solo Track events.

Tool List

Along the right edge of the Bar Editor Page (shown in Figure 7.1 at the beginning of this chapter) is a list of words used during editing. Some are mode switches, some open windows where parameters can be tweaked, and some are direct commands to Music-X that cause things to happen instantly. They offer the most often used functions in the Bar Editor and are provided next to the score window for easy access. We'll call these words "tool words" and the list, the "tool list".

In general, tool words are active when highlighted in white, and inactive when written in blue. Tool words are activated, deactivated, or used by clicking them.

The following 17 subchapters give an explanation of each tool word and any windows or parameters associated with that word.

SetSig

("f" from the keyboard)

Read this tool word as "Set Signature". **SetSig** is used when editing sequences containing Time Signature Change events to resize the measure and beat lines to the current time signature.

Normally, measure and beat lines are resized automatically while the sequence plays to conform with Time Signature Change events as they are encountered. When the sequence is stopped, the measure and beat lines stay sized to the last valid time signature. If you then manually scroll to a portion of the sequence with a different time signature, the measure and beat lines may appear incorrect. Playing from that point would resize the lines, but if you don't want to play the sequence, you can use **SetSig** to resize the measure and beat lines to the new time signature.

If there are multiple Time Signature Change events currently in the window, then **SetSig** uses the last Time Signature Change event before the middle of the window as the current time signature when resizing measure and beat lines.

[see Bar Editor / Graphic Score Window / Measure Lines]

[see Bar Editor / Event Types / Time Signature Change]

Time Signature Display

Under the SetSig tool is a display showing the current time signature. This display is automatically updated as Time Signature Change events are encountered in a sequence. If there are no Time Signature Change events in the sequence, then the display shows the global time signature.

[see Bar Editor / Event Types / Time Signature Change]

This display is an indicator only and cannot be edited directly.

♦ **Note:** The global time signature can be changed from the Sequencer Page, or from the **Params** window in the Bar Editor, or by inserting a Time Signature Change event into the sequence.

[see Sequencer Page / Time Signature Window]

[see Bar Editor / Params / Time Signature]

Add

("A" from the keyboard)

Clicking on **Add** puts the Bar Editor into "Add mode". Add mode makes it possible to put new events into the sequence using the mouse.

While in Add mode, the word **Add** stays highlighted in white. While you are in Add mode, you can also grab, drag, and edit pre-existing events of any type.

When you first click **Add**, a requester appears whose title asks **What Type of Event?**, and offers a list of the possible event types that can be added to the score using the mouse.

Choose an event type by clicking on its name in the list. The chosen event type will appear highlighted in white. After you choose an event type, click on the **OK** button (or press the "O" key) to close the window and return to the editor.

While in Add mode, you can add any number of events of the chosen type to the sequence by clicking the mouse in an empty area of the graphic score window. Each time you add a new event, that event becomes the current event, and the virtual sliders or other applicable controls may be used to edit it before adding another event.

Here's how to make sure the event goes where you want it to go: Each time you click the left mouse button, a new event will appear attached to the mouse pointer. Before releasing the mouse button, drag the event into place. When you release the button, the event is entered into the sequence. If you make a mistake, just grab the event again and drag it to the right spot.

If the type of event being added is one that can have a duration (Note, Play Sequence, Tempo Change, etc.) then its default duration will be one clock less than

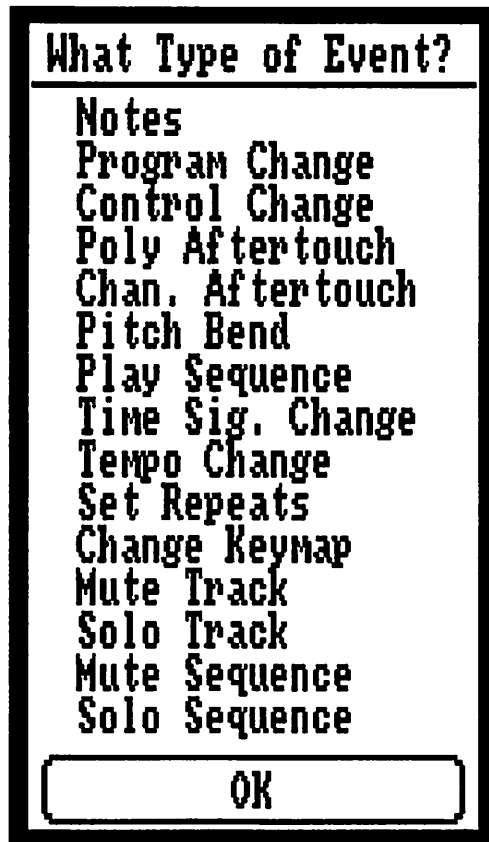


Illustration 7.9: Event Type Requester

that of the current duration value, as set in the Grid requester. The reason for this one-clock difference is to assure that the MIDI Note Off of a previous event is sent before the Note On of the next event. This assures the efficient use of voices when playing sequences on multi-timbral sound modules.

◆ Note: If you wish, you can override this feature, when setting the current Duration in the Grid requester, by increasing the chosen duration by one clock before closing the requester. **Add** will then subtract the extra clock, leaving the intended duration.

[see Bar Editor / Grid]

◆ Note: Adding an event type to the score turns on the display of that event type if it had been previously turned off in the display menu.

[see Bar Editor / Display Menu]

Insert

("I" from the keyboard)

Clicking on **Insert** puts the Bar Editor into "Insert mode". Insert mode is similar to Add mode; it makes it possible to put new events into the sequence using the mouse. The difference is that an amount of time equal to the current Duration, as set in the Grid requester, is inserted into the sequence along with each new event.

The tool word **Insert** stays highlighted in white while the editor is in Insert mode. While you are in Insert mode, you can also grab, drag, and edit pre-existing events of any type.

When **Insert** is first clicked, the **What Type of Event?** requester is called. This window has been described above under **Add**. Use this requester to choose the type of event to be inserted. After you choose an event type, and exit the requester, you are in Insert mode. Each time you click the left mouse button in an empty portion of the graphic score window you'll cause a new event of that type to appear attached to the mouse pointer. Before you release the button, you may move the mouse to drag the event to where you want it. The event will be inserted into the score when the mouse button is released.

When you are in Insert mode, any events entered with the mouse will push all following events back in time (later) to make room for the new event. The amount of time inserted is equal to the current Duration, as set in the Grid requester. No time is inserted for events with no duration (as with Pitch Bend).

As with Add mode, the durations of new inserted events are actually one clock less than the current Duration setting, for voice efficiency when using MIDI sound modules.

[see Bar Editor / Grid]

◆ Note: Inserting an event does not change the duration of any events that would be currently playing at the insertion point. In other words, any pending Note Offs are not pushed back. Inserting a short note in the middle of a long note will not make the long note longer.

Move

("C" from the keyboard)

Clicking on **Move** puts the Bar Editor into "Move mode". Move mode is used to drag, or edit pre-existing events in the score. Normally, if you are in **Add** or **Insert** mode, you can drag events too. However, **Move** is used when you want to avoid the possibility of accidentally entering a new event by clicking on a blank area of the score.

While in Move mode, the word **Move** stays highlighted in white.

To drag an event, click on it and hold the left mouse button. The event will become the current event and appear hollow or reversed. As long as you hold the mouse button, the event will be attached to the mouse pointer. Moving the mouse pointer moves the event as well. Once an event becomes the current event, the virtual sliders and other controls can be used to edit its parameters.

Remove

("V" from the keyboard)

Clicking on **Remove** puts the Bar Editor into "Remove mode". Remove mode is used to remove or erase pre-existing events from the sequence one by one.

The tool word **Remove** stays highlighted in white as long as Remove mode is on.

While in Remove mode, clicking on any event in the graphic score window will remove it from the score.

If you remove an event accidentally, you can restore it by pressing the HELP key on the Amiga keyboard. Repeatedly pressing the HELP key will restore up to 16 events previously removed from the sequence.

◆ Note: Changing between editors resets the HELP key. You must restore accidentally removed events before switching editors.

Lock

(SHIFT-"L" from the keyboard)

Clicking on **Lock** puts the Bar Editor into "Lock mode". Lock mode is used in conjunction with the Channel Square to hide some of the events in a sequence so you can more easily edit the events you're interested in. This is handy when working with composite sequences, containing data on many MIDI channels.

The tool word **Lock** stays highlighted in white while the editor is in Lock mode. While you are in Lock mode, you can also grab, drag, and edit pre-existing events of any type.

While **Lock** is on (highlighted), one can click on various channels in the channel

square (any except the current channel) to lock them out of the bar graph display. When a channel is locked, its number is shown in red in the Channel Square. Any events on a locked-out channel are heard but not seen. Clicking again on a locked channel unlocks it and lets any events on that channel be displayed again.

[see Bar Editor / Channel Square]

Example: Locking out all but one channel.

- ☐ Click **Lock** to enter Lock mode. **Lock** will appear highlighted.
- ☐ Click on an event in the score whose channel you wish to see alone. The Channel Square will show that channel in white.
- ☐ Click on all the other channels in the Channel Square to lock them out. (They will turn red.)
- ☐ Now only events on the current channel will be displayed and can be edited.

To restore all the channels, click all the reddened channels in the Channel Square. A quick trick that restores all locked channels is to choose **Bar** from the edit menu. Since we are already in the Bar Editor, this function will simply reinitialize the screen. Part of the reinitializing process is to unlock all channels.

◆ Note: Various event types can also be temporarily hidden using the Display menu.]

[see Bar Editor / Display Menu]

Select

("E" from the keyboard)

Clicking **Select** puts the Bar Editor into "Select mode". The mouse then becomes a tool for drawing boxes around events or groups of events to "select" them for future group operations.

To select an event or group of events, simply click and hold the left mouse button anywhere in the score window to define one corner of the box. Now drag the pointer until you see the outline of a box appear and expand. When you release the mouse button, any events within the box will become selected, and the box will disappear. Selected events will change in appearance to show that they've been selected. Some, like Note events, appear hollow when selected. Others use reversed colors.

It's possible to draw boxes bigger than can be displayed on the screen at one time. Simply move the mouse off the edge of the graphic score window while dragging the corner of the box and the screen will auto-scroll to follow you. You'll find it quicker work to zoom out before selecting events in large sections of time.

Selected events are cumulative. Selecting a new event does not de-Select other events. This is handy when selecting individual notes out of a cluster, as when picking out the inner voice of a trio where the three parts overlap in range. One can move through the score, selecting a few notes here and a few notes there until all the desired notes are selected. (Since selection is cumulative, it's a good idea to use **Unmark** before selecting so that no previously selected events are included in the group accidentally.)

[see Bar Editor / Unmark]

◆ Note: It's not necessary to box a complete Note event to Select it. But you must box at least the head of the note. This will come in handy when Selecting notes with long durations.

After a group of events has been selected, many other editing functions can be used to edit them collectively. For instance, clicking on the MIDI Channel Square will change their MIDI channel assignments. Pressing the CursorUp and CursorDown keys will transpose all selected (and transposable) events by halfsteps. Pressing the CursorLeft and CursorRight keys will move all the selected events through time in increments of the current Grid Size when **Snap** is on, and in increments of one clock when **Snap** is off. The cut and paste operations available in the Options menu are all used in reference to a group of selected events. The Quantizer, Velocity Scaler, and Aftertouch Scaler modules can all be used to operate on selected events.

[see Bar Editor / MIDI Channel Square]

[see Bar Editor / Grid]

[see Bar Editor / Snap]

[see Bar Editor / Options Menu]

[see Bar Editor / Modules Menu]

Mark

("M" from the keyboard)

Clicking **Mark** puts the Bar Editor into "Mark mode". Mark mode is similar to Select mode in that it defines a group of events for use in group operations. The difference is that Mark mode deals with regions of time and the events within that time, while Select mode only deals with the events. This means that **Mark** can be used in conjunction with cut and paste operations to splice sections of musical time together, as when changing the order of measures in a sequence.

The tool word **Mark** stays highlighted in white while the editor is in Mark mode.

Here's how to Mark a region of time:

- ☐ Move the mouse pointer to the beginning of the section you wish to mark.

- ☐ Click and hold the left mouse button. A vertical magenta line will appear, marking the beginning of the region. The **T=** field, below the score window, will indicate the start time for the beginning of the region.
- ☐ Hold the mouse button and drag the pointer to stretch the magenta region to the end of the section to be marked. If the region to be marked is larger than the portion of the sequence shown on the screen, simply move the mouse pointer off the edge of the graphic score window while dragging, and the score window will auto-scroll to bring the needed portions of the sequence into view. The **D=** field, below the score window, will indicate the total duration of the marked area as it grows.
- ☐ When the pointer is positioned over the end of the region, release the mouse button. The region is now marked.

If you make a mistake in marking a region, click **Unmark** (see below) and start over.

An alternative method for marking large portions of time is to click once at the beginning point, then scroll forward or play the sequence to find the end of the section. Once the end is found, position the pointer over the end point, hold the **SHIFT** key, and click to mark the entire region between the two points.

◆ **Note:** When using **Mark** to move measures around, you'll probably find it handy to set the Grid Size to as large a value as practical and use **Snap**. Also when marking large portions of time, you'll find it quicker work if you zoom out (**Zoom-**) first, so a larger portion of the sequence can be seen in the window at once.

Unmark

("U" from the keyboard)

The **Unmark** tool word is a command that causes all currently selected events to become deselected, or a currently marked region of time to become unmarked. **Unmark** is used to "clean the slate" before using either **Select** or **Mark**, or after making a mistake while using either of these.

The word **Unmark** only stays highlighted while the events are being unmarked. Once they are unmarked, **Unmark** returns to normal. **Unmark** can be used while in any mode.

Zoom+ and Zoom-

(SHIFT-"Z" from the keyboard)

("Z" from the keyboard)

Read these as "Zoom increase" and "Zoom decrease," or "Zoom in" and "Zoom out". Use these tool words to change the time scale of the graphic window display. This works similarly to a zoom lens on a camera; a higher zoom gives a closer look at the subject, a lower zoom gives a wider look at the surroundings.

There are 7 levels of Zoom; each level shows half or twice the time scope of the neighboring Zoom level. Each time you click on **Zoom+** or **Zoom-**, the time scale displayed in the graphic score window is halved or doubled.

These tool words are direct commands and may be used in any mode. They stay highlighted temporarily, while the window is readjusted, then return to normal.

Increasing zoom (**Zoom+**) shows less actual time on the screen. This allows mouse manipulations of data in resolutions as small as one or two clocks. (Movements of exactly one clock can also be effected upon the current event, or selected events, by using the CursorLeft and CursorRight keys, while **Snap** is off.) Decreasing zoom (**Zoom-**) shows more time on the screen. This allows macro-scaled manipulations

(as when cutting and pasting song sections) by showing many measures on the screen at once.

[see Snap (below)]

◆ **Note:** The pitch scale doesn't zoom. It always stays in scale with the miniature keyboard graphic to the left of the graphic score window.

Snap

(SHIFT-"S" from the keyboard)

Snap is an optional mode switch, and may be used with any other mode. **Snap** was inspired by a similar function in CAD (Computer Aided Design) programs. It limits the placement of events in the score window to the nodes of an invisible "Grid". **Snap** is turned off and on by clicking it. When on, it appears written in white.

[see Bar Editor / Grid (below)]

Snap affects various editing procedures differently. The procedures that **Snap** affects are:

- ☐ Adding or Inserting new events into the score.
- ☐ Pasting the Clip into the score.
- ☐ Changing the start times of pre-existing events in the score (moving them).
- ☐ Changing the durations of pre-existing events in the score.
- ☐ Marking a region of time.

When Adding or Inserting new events, **Snap** restricts them to being placed on Grid

points only, rather than being placed freely. When **Snap** is off, new events can be placed anywhere in time, down to a resolution of one clock (1/192 quarter note).

When pasting the Edit Clip into the score using **Paste**, the start time of the first event in the Clip is limited to Grid points only. All following events in the Clip stay in their original relationship with the first event. For example, when pasting a Clip containing two events an eighth note apart while **Snap** is on, the first event will be restricted to Grid points and the second note will always be one eighth note later than the first, regardless of what the Grid size is.

[see Bar Editor / Options Menu / Paste]

When pre-existing events are moved about in the score, **Snap** allows them to be moved *in increments of the current Grid Size* (rather than being moved to Grid points specifically). That means that if the event was not “right on the beat” before being moved, it will still have the same error after being moved. This is good because events retain their original rhythmic feel, as far as being a little early or late, while being moved around in the score. This applies whether events are being dragged with the mouse or moved with the cursor keys. When **Snap** is off, the start times of pre-existing events can be nicely adjusted down to a resolution of one clock.

◆ Note: You can get some interesting rhythmic-offset effects with **Snap** by temporarily changing to an unusual Grid Size and moving a group of selected notes with the CursorLeft and CursorRight keys while **Snap** is on. Example: Select a group of 6 quarter-note triplets in a drum sequence. Change the Grid Size to eighth notes and turn **Snap** on. Use the CursorLeft key to move the selected triplet notes 1 eighth note earlier in the score. Off beat triplets can give you that Latin timbale-solo sound.

[see Advanced Users / Editing Tricks / Rhythmic Offsets]

When changing the durations of pre-existing events, **Snap** allows them to be changed *in increments of the current duration*. Example: If the current Duration is set to quarter notes, then an eighth note in your sequence could be stretched to

become a dotted quarter note (1 eighth plus 1 quarter), but not a quarter note. As you dragged its tail with the mouse, you would see it change suddenly (that's the "snap") from an eighth note to a dotted quarter note. That's because **Snap** prevented it from changing by any less than a quarter note's duration. This applies as well when using the cursor keys to change the durations of the current event or group of selected events. When **Snap** is off, the durations of pre-existing events can be nicely adjusted down to a resolution of one clock.

◆ Note: The current Duration is also set in the Grid requester.

When marking a region of time using **Mark**, **Snap** restricts the edges of the region to Grid points. As the region is dragged and expanded, it will appear to increase in size in jumps. Setting the Grid Size to maximum and turning **Snap** on before cutting a song section will keep the section's length to multiples of a measure and assure that when the section is pasted back into the score, events moved aside to make room for the new section will retain their original rhythmic identity. When **Snap** is turned off, regions of any size can be marked with one-clock accuracy.

Quiet

("Q" from the keyboard)

Quiet is an optional mode switch, and may be used with any other mode. **Quiet** turns off the audible feedback heard when Note events are dragged in the graphic score window.

While **Quiet** is on (highlighted), feedback is turned off. Click **Quiet** to turn it on and off. **Quiet** does not affect the playback of events when the editor is in play or record.

[see Bar Editor / Options Menu / Feedback]

An extra editing trick is possible when **Quiet** is turned on and Feedback is defeated. While the sequence is playing, a new event may be added or a pre-existing event

may be grabbed without stopping the clock. As long as you hold the left mouse button the event may be dragged to different positions. Instead of Feedback playing the event each time it is moved (as when **Quiet** is off) the event will be played each time the green Time line passes over it. If the sequence is short and repeating, different positions for the event can be tried and listened to as the sequence repeats. As soon as you release the event, by releasing the mouse button, the clock will stop and the event will be entered into the score at the current position.

Scroll

("L" from the keyboard)

Scroll is an optional mode switch, and may be used with any other mode. **Scroll** controls the auto-scroll function of the graphic score window. Click **Scroll** to turn it on and off.

If **Scroll** is on (highlighted), then when the editor is in play or record, the graphic score window will automatically scroll along through the score in musical time, showing new events approaching from the right and passing through the time line as they are played.

If **Scroll** is off, the score window will not automatically scroll as the sequence is played, but may still be manually scrolled. When you first click **PLAY**, the Time Line travels from the left edge of the screen and continues off the right edge of the window and disappears. Events being played off screen would be heard but not seen as the Time Line passes through them. You might prefer to turn **Scroll** off when working in high zoom (closeup) since the screen would otherwise rapidly scroll away from the area you want to see.

[see Bar Editor / Zoom+ and Zoom-]

Grid

("G" from the keyboard)

This tool word is used to call the Grid requester where the current **Grid Size** and **Duration** parameters may be altered.

The two parameters Grid Size and Duration are used by many other functions in the Editors. Grid Size and Duration are used by **Snap**. Grid Size and Duration are both used by the Quantizer Module. Grid Size is used by the **STEP** button when recording in step mode. Duration is used by **Add** and **Insert** to set the durations of new notes being entered into the score.

What is the Grid?

The Grid is an invisible structure of time points mapped over a sequence and used for reference when editing. The Grid works differently in Relative and Absolute sequences.

When editing a Relative sequence, Grid points are separated by a set number of clocks called the "Grid Size". Clocks are the smallest unit of resolution in Relative sequences and represent 1/192 quarter note.

When editing Absolute sequences, the smallest unit of resolution is 1/4 frame. The number of frames per second depends on the Time Code you are using. Time Code is set in the Sync menu. The graphic score window can only show a resolution of single frames visually, but the actual musical data is recorded and stored in quarter frames. (Example: With **Snap** off, move the current event with the CursorRight key. You'll notice that for every four key presses, the frame number in the **T=** field will increase by one.)

[see Sequencer Page / Sync Menu / Time Code]

[see Bar Editor / Snap]

[see Bar Editor / T=]

What is the Duration parameter?

The Duration parameter sets up an arbitrary duration that can be used as a model by other functions in Music-X. For instance, when adding new note events to a sequence in the Bar Editor, the **Add** function uses the current Duration parameter as the model duration for each new note. Also when changing the durations of pre-existing events in the sequence, while **Snap** is on, the current Duration parameter is used as model for the amount of change that can be applied to the durations of the pre-existing notes.

Clicking **Grid** opens the **SET GRID SIZES** window in the Bar Editor. Depending on the type of sequence currently being edited, one of two similar windows will appear. Take a look at the Grid requester window for Absolute sequences below.

The image shows a window titled "SET GRID SIZES". It contains two rows of controls. The first row is labeled "Grid Size:" followed by a box containing the value "0060" and a horizontal slider bar. The second row is labeled "Duration:" followed by a box containing the value "0120" and a horizontal slider bar. At the bottom center of the window is a button labeled "OK".

Illustration 7.10: Grid Requester (Absolute)

In this requester, you'll see two identical sliders titled **Grid Size:** and **Duration:**. The sliders range from 1 to 3600. The sliders are used to adjust the number of quarter frames for each parameter. Once the parameters are set correctly, the window may be closed by clicking the **OK** button or pressing the "O" key.

[see Bar Editor / Modules Menu / Quantizer Module]

[see Bar Editor / Edit Menu / Auto-Step]

The Grid requester used when editing Relative sequences contains slightly expanded controls. Besides the two sliders, there are displays of musical note values including a "dot", and some tuplet numbers. The sliders represent the number of

clocks in the current **Grid Size:** and **Duration:** parameters. They range from 1 clock to the number of clocks in one measure. Example: When the Time Signature is 4/4, the sliders range from 1 to 768 clocks. The maximum slider range is 1 to 3072 clocks with a Time Signature of 16/4.

As you drag either slider, the number written on the slider itself will change to show

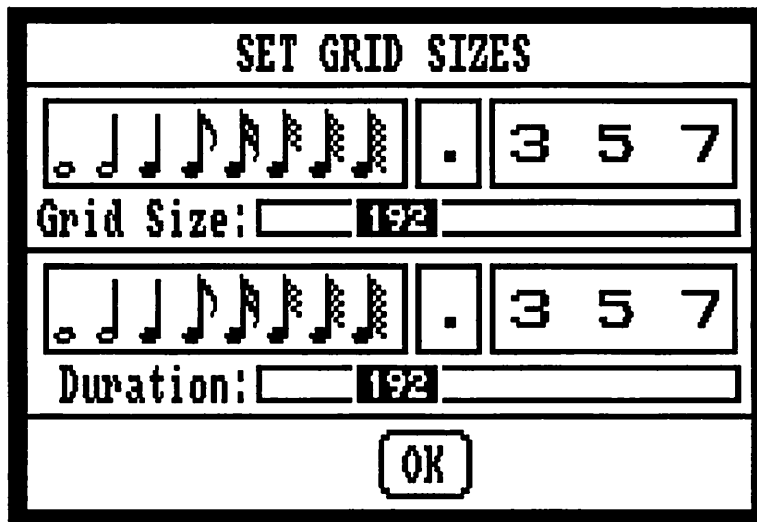


Illustration 7.11: Grid Requester (Relative)

the number of clocks in the parameter. As the number of clocks increases or decreases, the closest musical note value will be highlighted in purple. If the clocks are exactly the correct number for a musical note value, that note graphic is highlighted in white instead of purple.

You can also click directly on the note values, the dot, and the tuplet numbers themselves. When you do, the slider moves to the correct number of clocks in that note value. The tuplet numbers are mutually exclusive. Clicking on any one of them turns the others off. If you don't want any tuplets, clicking on the currently highlighted tuplet number turns it off. The dot is turned on and off by clicking on it as well.

The dot and tuplet numbers work as in normal music theory. Adding a dot to a note increases its duration to $3/2$ of normal. It adds half of the original value. (Example: A quarter note contains 192 clocks; a dotted quarter note contains half again as many, 288 clocks.) Triplets (noted by the number **3**) are worth $2/3$ normal value. (Example: A quarter note contains 192 clocks; a quarter note triplet contains 128 clocks.) Quintuplets (noted by the number **5**) are worth $4/5$ normal value. Septuplets (noted by the number **7**) are worth $4/7$ normal value.

If you want to use other polyrhythms, besides the ones shown in the requester, you can do so by figuring out how many clocks are in one of your beats and set it up using the slider.

Example: To set up a custom Grid Size of 11 notes per measure of 4/4, for quantizing or editing:

- ☐ A whole note (768 clocks) divided by 11 notes is 69.82 clocks per note. Round off 69.82 to 70 clocks.
- ☐ From within the Bar Editor, click **Grid** to open the Grid requester. Set the **Grid Size**: slider to 70 clocks (slightly less than a dotted 16th).
- ☐ Close the Grid requester by clicking **OK**.
- ☐ Turn **Snap** on (click it so it's highlighted). Turn **Add** on (click it so it's highlighted), and add a measure of 11th notes in the score window with the mouse to see that the Grid is working.
- ☐ The Grid can now be used to Quantize as well.

A bit about accumulated errors and error correction: 11 notes of 70 clocks each total 770 clocks. Our measure of 4/4 is only 768 clocks. The next (12th) Grid point would be $770 - 768 = 2$ clocks too late. However, the Grid automatically resets to the beginning of each measure so that accumulated errors are corrected. The Grid starts again at 768 clocks; our polyrhythm stays synchronized.

◆ Note: polyrhythms that divide evenly into the number of clocks per measure have no accumulated errors and need no error correction.

Params

(SHIFT-“P” from the keyboard)

This tool word is used to open the **PLAY PARAMETERS** window where various parameters governing the way the sequence is played may be altered. Some of these parameters are also found in the Sequencer Page but have been made available here for convenience.

PLAY PARAMETERS		Tempo: 100	Memory: 10886
☐ Enable Sync	☐ Remap T/S	4 4	OK
☐ Play in Context			
Start: 0000			

Illustration 7.12: PLAY PARAMETERS Window

Descriptions of the various parameters available from this window are listed below.

Tempo:

At the top of the window is a tempo slider. This slider works exactly like the tempo slider in the Sequencer page and affects the global tempo setting used by the sequencer and editors during playback.

This slider can be used to audition a sequence at various tempos during editing. If you click **PLAY** before calling the Params window, the sequence will continue to play while the window is open. You'll be able to adjust the tempo and hear the changes while the sequence is playing, even though the background screen will temporarily appear frozen during window access.

◆ Note: If there are any imbedded tempo events in your sequence they will take precedence over the global tempo setting shown here.

Memory:

To the right of the tempo slider is the **Memory:** readout. This is an indicator of available free memory. It shows an estimation of the number of Note events (not bytes) that could still be added to memory before running out of room. The average Note event uses 10 bytes: 3 to store the start time, 3 to store the MIDI data for a Note On, 3 to store the stop time, and 1 for the release velocity. This readout is the same as the one found in the Sequencer Page.

[see Sequencer Page / Tempo Window / Memory:]

◆ Note: This indicator may show a slightly different number than the one in the Sequencer Page. This is because different amounts of memory overhead are required by the different pages, The available free memory actually changes slightly from page to page.

Enable Sync

The **Enable Sync** switch is used to allow the editors to play along with a drum machine in the same manner as the Sequencer Page would. This can be useful for hearing a sequence “in context” using a separately programmed drum machine with the sequencer.

Playing a sequence from within the editors normally uses the Internal clock if a Relative sequence (one based on measures, beats, and Clocks) is being edited, and Video Clock if an Absolute sequence (based on Hours, Minutes, Seconds, and Frames) is being edited. The **Enable Sync** switch in the Params window overrides the default setting and enables the use of MIDI Clocks, MIDI Start, MIDI Stop, MIDI Continue, and Song Position Pointer messages as indicated by the current settings in the Sync menu. This basically enables the drum machine.

[see Sequencer Page / Sync Menu]

Play In Context

The **Play In Context** switch is used to hear the currently edited sequence as it will ultimately sound with the rest of the performance. This is good for listening to see if the edits being made are musically compatible with the other sequences in memory. When the **Play In Context** button is turned on, the Editor Pages will Play In Context whenever the **PLAY** button is clicked. **Play In Context** cannot be used when Step recording.

When playing in context, the entire performance will play as though playing from the Sequencer Page. The current sequence being edited will only be heard if self starting (if not turned off in the Sequencer Page) or if triggered by another sequence using a Play Sequence event. Each time the current sequence is triggered, or starts by itself, the sequence will be heard and (if **Scroll** is on) the Editor will scroll to show the portion of the sequence being played.

[see Sequencer Page / Sequence List / << >> (Offset Arrows)]

[see Bar Editor / Scroll]

◆ Note: If you are using a drum machine as a virtual sequencer (you have patterns programmed into the drum machine rather than triggering it from the sequencer) you'll also want to use **Enable Sync** (see above).

Start:

Read this as "Start Offset".

The **Start:** slider is used in conjunction with the **Play In Context** switch to let the performance start from different positions. This is similar to changing the clock positions in the Sequencer Page before clicking **PLAY**. If the **Play In Context** switch is not on, the **Start:** slider is ignored.

Normally, when playing in context, the performance starts from the beginning of the song and continues to the end. This is good for hearing the whole piece as it will ultimately sound with the currently edited sequence. However, when editing, you

may wish to listen to only a certain portion of the performance to see how the edits are coming along. This slider can be used to set the number of measures that will be skipped past the beginning of the song before starting to play the performance. The slider ranges from 000 to 500 measures. When the slider is set at 000 measures this means that no measures will be skipped and the song will start from the beginning each time **PLAY** is clicked. Setting the slider at 001 measures means that one measure will be skipped before playing the song; the song will start on the second measure.

For example, suppose you are editing a piano sequence containing a solo that is supposed to occur at the 32nd measure of the song, and you want to hear what it will sound like with the other sequences. Turn on **Play In Context**, and set the **Start:** slider to 31 measures. When the **PLAY** button is clicked in the Editor, the song will start playing from the 32nd measure and the current sequence will play along with it.

Remap T/S

Read this as “Remap Time Signature”.

This switch is used in conjunction with the Time Signature control (see below) to change the metric structure of a sequence.

There are two approaches to changing the Time Signature of a prerecorded sequence.

The first approach is to expand or contract the measure lengths by adding or subtracting beats from the end of each measure. This is the default method in Music-X. The advantage to this is that measure integrity is maintained. In other words, the same notes still occur within the same measures after the Time Signature change. The disadvantage to this is that when changing to a smaller measure size, notes at the end of each measure would seem to clump together.

The second approach is to leave the music alone and simply map a new Time Signature over top of it. The music sounds the same but is structured differently.

The **Remap T/S** switch allows you to use the second approach. To remap the current sequence into a new time signature, turn the **Remap T/S** switch on, then change the Time Signature control (see below) to the desired Time Signature, then close the Params window by clicking **OK**.

◆ Note: When remapping a sequence containing Time Signature Change events, the **Remap T/S** function takes into account the way the music sounded with the Time Signature Changes in it. After remapping, those Time Signature Change events are unneeded and should be manually removed from the sequence.

Time Signature Control

You can change the global time signature with this control, as you would in the main Sequencer Page. Click on the green arrows to increase or decrease the numerator or denominator of the Time Signature. This globally affects all sequences.

◆ Note: Any TSIG events in the currently playing sequence(s) will override the global Time Signature.

[see Sequencer Page / Time Signature Window]

OK

Click here when done, to close the **PLAY PARAMETERS** window and continue editing.

Repeat

(SHIFT-“R” from the keyboard)

The last of the tool words in the Tool List is **Repeat**, found directly below the word **Params**. **Repeat** is an optional mode switch, and may be used with any other mode. **Repeat** controls the auto-repeat function of the graphic score window. Click **Repeat** to turn it on and off.

This word is normally turned on (highlighted in white) by default. When on, it causes the sequence to auto-repeat *when in the editor only*. When the green Time line reaches the yellow End line during play, the clock returns to the beginning of the sequence and starts to play it again. This is usually handy when editing, especially when working on small looping sections, as when writing drum parts. Conversely, if you're working near the end of a long sequence, you may wish to turn this off to prevent the score window from jumping to the beginning when it reaches the end of the sequence.

◆ **Note:** When Playing In Context, the auto-repeat function is defeated to give true playback.

[see Bar Editor / Params / Play In Context]

◆ **Note:** If you want the sequence to repeat when played from the Sequencer Page, you must Add a Set Repeats event.

[see Bar Editor / Add]

[see Bar Editor / Event Types / Set Repeats]

Seq:

Read this as “current sequence number”. This indicator is found just above the **PAUSE** button, against the lower left edge of the Bar Editor Page. When you first enter the Bar Editor by clicking **EDIT** in the Sequencer Page, the number of the sequence to be edited is written to this field.

The current sequence number is the sequence that will be overwritten when you exit the editor and **STORE** the edit buffer to sequence memory.

[see Bar Editor / File Menu / Exit]

The current sequence can be changed in two ways. First, by bringing a new sequence into the editor using **Select...**, in the File menu. Second, by storing the current sequence to other than the original location using **Store...**, in the **File** menu.

[see Bar Editor / File Menu / Select...]

[see Bar Editor / File Menu / Store...]

T=

This indicator is found just above the transport controls. Think of it as the “Time equals” indicator.

The T= indicator has five uses. First, it shows the starting time of the current event when grabbed or dragged. Second, during a paste operation, when pasting a clip of previously selected events, it shows the starting time of the first event in the clip as it is being placed in the score. Third, during a paste operation, when pasting a region of previously marked time, it shows the point at which the clip will be inserted into the score while it is being positioned. Fourth, when the sequence is playing it shows the current clock time. Fifth, when there is no current event, it shows the time under the mouse pointer.

[see Bar Editor / Options Menu / Paste]

[see Bar Editor / Select]

[see Bar Editor / Mark]

For Relative sequences, this indicator reads out in measures, beats, and clocks. For Absolute sequences, it reads in hours, minutes, seconds, and frames.

D=

This indicator, found just above the transport controls and to the right of the T= indicator, can be thought of as the “Duration equals” indicator. It’s used to display the durations of various things during editing. In certain cases, it’s also used as a transposition indicator.

This indicator has three uses as a duration indicator. First, it shows the duration of the current event. Second, while in Mark mode, it shows the duration of a region of time being marked. Third, while in Select mode, it shows the time inside the Select box as the box is extended.

When being used as a duration indicator, this indicator reads out in measures, beats, and clocks for Relative sequences, and hours, minutes, seconds, and frames for Absolute sequences.

During paste operations, this indicator is temporarily used to show the transposition of the clip when it’s being positioned in the score. In this case it reads out in the number of halfsteps away from the original key.

[see Bar Editor / Mark]

[see Bar Editor / Select]

[see Bar Editor / Options Menu / Paste]

Transport Controls

The transport controls in the Bar Editor work similarly to the transport controls in the Sequencer Page. However, there are no fast forward or rewind buttons in the Bar Editor. Instead, the score window itself is repositioned using a scroll bar. Following are descriptions of the Transport Controls in the Bar Editor.

*Illustration 7.13: Transport Controls***PAUSE**

(SPACE BAR from the keyboard)

PAUSE is used to temporarily halt the clock while playing or recording.

If **PAUSE** is used during playback, currently playing notes are temporarily held, and finish naturally when **PAUSE** is released.

If **PAUSE** is used while recording, the clock stops but the recording process doesn't. This feature is used with the **STEP** button to implement step recording.

[see Bar Editor / Recording Modes (below)]

PLAY

("P" from the keyboard)

(ESC (Escape key) = All notes off)

(SHIFT-ESC = Super All notes off)

In the Bar Editor, clicking **PLAY** will locate the clock to the beginning of the portion of the sequence currently shown in the window and begin playing from there. In other words, the clock will start at (and the green Time Line will scroll from) the left edge of the window.

If at anytime during play, a "MIDI stuck-on note" happens, remember that pressing the Amiga's Escape key (at the upper left corner of the keyboard) sends a MIDI "All Notes Off" message. Holding the SHIFT key while pressing the escape key initiates "Super All Notes Off", which actually sends a MIDI Note Off message for each of the 128 possible notes on all 16 MIDI channels.

RECORD

("R" from the keyboard)

There are three methods for recording in the Bar Editor: Live Recording, Step Recording, and Step Recording with Auto-Step. These methods are discussed in the next subchapter.

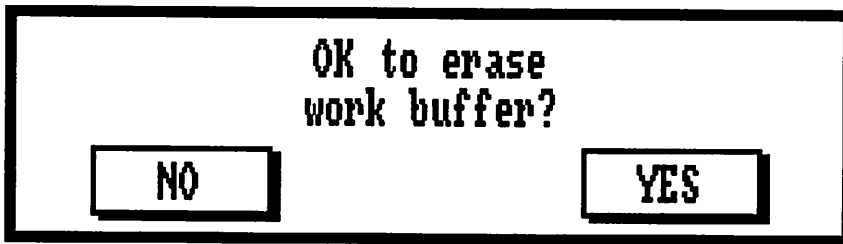


Illustration 7.14: Record Safety Prompt

If the Edit Buffer is not empty when you click the **RECORD** button, a safety prompt will come up to prevent accidental erasure. Click **NO** to back out safely. Otherwise click **YES**, or press the RETURN key to begin recording.

STOP

("S" from the keyboard)

Clicking the **STOP** button halts the clock and stops any currently playing notes. Clicking **STOP** while in record, adds an End event into the sequence at the current clock position.

Recording Modes

There are three ways to record in the Bar Editor: Live recording, Step recording, and Step recording with Auto-Step. Each method has its strengths. The three methods are discussed below.

Live Recording

Live recording in the Bar Editor is basically the same as the recording process in the Sequencer Page. However, the options have been scaled down. You cannot Punch-In, there are no Count-In Bars, etc. You essentially click **RECORD** and begin playing the mother keyboard. The snazzy thing about recording live in the Bar Editor is that you can see the notes appear as they are recorded.

Example: Recording a drum fill sequence from within the editor.

- ☐ Click **Params**. Then from within the **PLAY PARAMETERS** window, turn on the **Play In Context** switch, and drag the **Start:** slider to zero. Click **OK** to close the window.

[see Bar Editor / Params / Play In Context]

- ☐ Turn on **Scroll** in the Tool List.
- ☐ Manually scroll the Graphic Score Window to the beginning of the sequence..
- ☐ Click **RECORD**. If the sequence is not currently empty, a safety prompt will appear to prevent accidental loss. Click **NO** to back out safely. Otherwise, click **YES**, or press the RETURN key to proceed.
- ☐ The Record Buffer will be emptied, and the editor will begin recording. Other sequences in memory can be heard playing along since we are using **Play In Context**.
- ☐ As the parts of the song come up that need drum fills, play the part on the mother keyboard or MIDI drum pad.
- ☐ As each note is received, it will appear in the graphic score window. (Attack and Release velocities are not displayed during recording, to speed up the display.)

- ❑ When done, click **STOP**. The sequence may now be edited and eventually stored.

Step Recording

Step recording allows you to manually control the movement of the clock during record using the **STEP** and **BACK** buttons. That means you can take as much time as you need while deciding what notes to play next. It also means you can back up if you make a mistake, and correct it right away before going on. This can be handy for non-keyboard players. It can also be handy for recording difficult or impossible parts. For instance, a moving horn section, with spread voicings in 16th notes.

Step recording is implemented with the help of the **PAUSE** button, and the **STEP** and **BACK** buttons. The idea is, if **PAUSE** is clicked before **RECORD**, the editor begins recording but the clock doesn't move. This is the beginning of manual control of the clock. Once the clock is stopped, you can take your time about placing your fingers on all the right notes. The **STEP** button then comes into the picture to move the clock ahead in controlled increments. Each time **STEP** is clicked, the clock moves ahead by the amount of clocks in the current Grid Size parameter. In other words, you're stepping along the Grid points, one at a time. By pressing keys at your leisure, and then advancing the clock, you can put together music of difficulty beyond your own ability to play.

[see Bar Editor / Grid]

Working with Durations

Each time a note is pressed or released, that action gets recorded at the current clock position. If you press and release the same keys at the same clock position, the new notes will have no duration. The way to do it is; press and hold a key, advance the clock, then release the key. The more times you advance the clock while the key is held, the longer the resulting duration will be.

The **BACK** button is used if you make a mistake, to erase the last note played. Each time you click the **BACK** key, the most recent note is erased. Clicking **BACK**

repeatedly continues to erase notes. As notes are erased, the clock is also rewound to the point at which the last erased note was before it was erased.

◆ **Note:** If your mother keyboard is set up away from the Amiga, or you just find it awkward to use the mouse while recording, a Keymap can be used to set up one or more notes on the mother keyboard (the bottom or top notes are usually the handiest) to trigger the Music-X command **Time Step** which is the equivalent of clicking the **STEP** button in the Editor Pages.

[see Keymap Editor / Action List / Music-X Command / Time Step]

Example: Using Step Mode to record a walking bass part.

- ☐ Use the **SET GRID SIZES** window to set the Grid Size to quarter notes.
- ☐ Manually scroll the sequence to the beginning.
- ☐ Click **PAUSE** to freeze the clock.
- ☐ Click **RECORD**. If the sequence is not currently empty, a safety prompt will appear to prevent accidental loss. Click **NO** to back out safely. Otherwise, click **YES** or press the RETURN key to proceed.
- ☐ The Record Buffer will be emptied, and the editor will begin recording. The clock will not move however, since **PAUSE** is on.
- ☐ On the mother keyboard, press and hold the first note of the bass line. We've just recorded a Note On at the beginning of the sequence and we are sustaining the note by holding it.
- ☐ Click **STEP**, or press the "]" (right bracket) key, to step the clock forward by one quarter note. The note now has some duration, and will appear in the score window.

- ☐ Release the note on the mother keyboard. We've just recorded a Note Off on beat two of the sequence, creating a full Note event with a duration of one quarter note.
- ☐ Repeat the last three steps for each note in the bass line. (Press the note, advance the clock, release the note.)
- ☐ If you make a mistake, click **BACK**, or press the "[" (left bracket) key, to erase the previously recorded note.
- ☐ To record a rest, advance the clock by clicking **STEP**, or pressing the "]" (right bracket) key, without pressing a note.
- ☐ To record a note with a longer duration than the current Grid Size, press the note, advance the clock more than once, and release the note.
- ☐ To record a note of shorter duration than the current Grid Size, open the **SET GRID SIZES** window while in record, and set the Grid Size to the desired duration, then continue recording.

Step Recording with Auto-Step

Auto-Step mode is enabled by choosing **Auto-Step** from the Edit menu of the Bar Editor. When the menu item appears checked, Auto-Step mode is enabled.

[see Bar Editor / Edit Menu / Auto Step]

Auto-Step mode makes it even easier to enter notes when Step recording. Auto-Step causes the clock to advance forward in increments of the current Grid Size each time a note is released on the mother keyboard. This means it is not necessary to click the **STEP** button each time you want the clock to move ahead.

Durations are determined in Auto-Step mode using the current Duration parameter

setting, as set in the Grid requester. What happens is, when you first press a key on the mother keyboard, the beginning of that note is recorded at the current clock position. When you release the key, that note is assigned the same duration as the current Duration parameter set in the Grid requester, and the clock is stepped forward by the amount of time in the current Grid Size parameter, also set in the Grid requester. The current Grid and Duration sizes may be different. By changing the Grid or Duration size while in record mode, one can build a melody with many complex rhythms.

Example: Using Auto-Step to record a difficult melody:

- ☐ Make sure Auto-Step is enabled in the Edit menu (the menu item should appear checked).
- ☐ Click the tool word **Grid** to open the Grid requester, and set the Grid Size and Duration to the value of the first note in the melody. (If you are going to play a sixteenth note, then set both parameters to sixteenth notes.)
- ☐ Click the **PAUSE** button to freeze the clock. It will appear highlighted.
- ☐ Click **RECORD**. It will appear highlighted.
- ☐ If the Edit Buffer is not currently empty, a safety prompt will appear to prevent accidental loss. Click **NO** to back out safely. Otherwise, click **YES** to proceed.
- ☐ Press and release the first note in the melody.
- ☐ The note will appear in the editor, and the clock will move forward.
- ☐ Open the Grid requester again and set the Grid Size and Duration to the value of the second note in the melody. (If it's the same value then you don't need to reset it.)

- ☐ Press and release the second note in the melody.
- ☐ If you make a mistake, click **BACK** and play the note again.
- ☐ Repeat the previous two steps until the melody is finished. Then click the **STOP** button to stop recording.

Chords can also be recorded in Auto-Step mode, as long as all the keys of a chord are depressed before any one of them is released.

Example: Using Auto-Step to record a difficult chord passage.

- ☐ Make sure Auto-Step is enabled in the **Edit** menu (the menu item should appear checked).
- ☐ Click the tool word **Grid** to open the Grid requester, and set the Grid Size and Duration to sixteenth notes.
- ☐ Click the **PAUSE** button to freeze the clock. It will appear highlighted.
- ☐ Click **RECORD**. It will appear highlighted.
- ☐ If the Edit Buffer is not currently empty, a safety prompt will appear to prevent accidental loss. Click **NO** to back out safely. Otherwise, click **YES** to proceed.
- ☐ Press and hold all the notes of the first chord, one by one, until all the chord is complete. Do not release any notes until the chord is complete, or the clock will move forward and all the notes will be terminated.
- ☐ Release the notes of the first chord. The chord will appear in the editor, and the clock will move forward.

- ☐ Use the Grid requester to change the current Grid Size and Duration if needed.
- ☐ Press and hold all the notes of the second chord, one by one, until all the chord is complete.
- ☐ Release the notes of the second chord. The chord will appear in the editor, and the clock will move forward.
- ☐ Repeat the previous three steps until all the chord passage is finished. Then click the **STOP** button to stop recording.

Channel Square

At the bottom of the Bar Editor Page is a small matrix consisting of the numbers 1 through 16. This is the MIDI Channel Square. It has three basic functions.

01	02	03	04
05	06	07	08
09	10	11	12
13	14	15	16

Illustration 7.15: Channel Square

The first function is to indicate and change the MIDI channel number of the current event. When a new event is grabbed, the channel square will highlight the channel number that event is on in white. Choosing other MIDI channels, by clicking on the Channel Square, will change the channel number for that event.

The second function is to change the MIDI channel numbers of a group of selected notes. After selecting one or more events with the **Select** function, click on a

channel number in the MIDI channel square. All the selected events will take on the new MIDI channel assignment.

[see Bar Editor / Select]

The third function of the Channel Square is as a selection tool when using the **Lock** function. When **Lock** is highlighted, the channel square is used to set which channels will be locked out of the display. This can be handy when editing composite sequences containing events on many MIDI channels. Locked channels are shown in red in the Channel Square. Events on a locked channel are heard but not seen. Clicking on a locked channel in the Channel Square unlocks it.

[see Bar Editor / Lock]

◆ Note: System Exclusive events and Music-X events (events that control the sequencer internally) don't use MIDI channels, and are not affected by Channel Square operations.

STEP and BACK

("]" from the keyboard)

("[" from the keyboard)

Just to the right of the Channel Square are two buttons labeled **STEP** and **BACK**. These are used during Step recording to advance or rewind the clock. The editor must be in record mode, with the **PAUSE** button on.

STEP and **BACK** are designed to be used during Record only and have no audible effect during Play or when the clock is stopped.

When **STEP** is clicked during Step record, the clock will move forward in increments of the current Grid Size, as set in the **Grid** requester. For example if the current Grid Size is set to a sixteenth note the **STEP** button will advance the clock by one sixteenth note (48 clocks) each time it is clicked.

When **BACK** is clicked during Step record, the last note recorded will be erased, and the clock will be moved back to the beginning of that note. If the last note erased is one of multiple Notes with the same start times, the clock will not be moved back until all of them have been erased by clicking the **BACK** button repeatedly. This can be handy during Step record when adding in the notes of a chord one by one. If you make a mistake, click **BACK** to erase only the most recently played note, and try it again.

Use the Grid requester to set the Grid Size and determine how much the clock will move each time **STEP** is clicked.

[see Bar Editor / Grid]

◆ Note: Using a Keymap, one or more keys on the mother keyboard can be set up to trigger the Music-X command **Time Step** which is equivalent to clicking the **STEP** button. The clock can also be automatically advanced in response to notes played on the mother keyboard during record, using **Auto-Step** mode.

[see Keymap Editor / Action List / Music-X Command / Time Step]
[see Bar Editor / Edit Menu / Auto-Step]
[see Bar Editor / Recording Modes / with Auto-Step]

Virtual Sliders

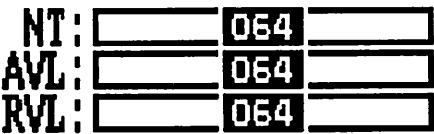


Illustration 7.16: Virtual Sliders

In the bottom right corner of the Bar Editor Page are three virtual sliders. These sliders have a number bar where most sliders have a drag bar. This means they can be used to both indicate a value, and change that value.

In the Bar Editor, these sliders are provided to give quick access to some of the parameters in the current event. Each time a new event is grabbed, the sliders reconfigure themselves to the parameters of that kind of event. Their names and ranges change automatically. The basic process when editing events is to first click on an event, then adjust the sliders.

♦ **Note:** Detailed descriptions of the possible configurations of these sliders are given in the descriptions of the event types.

[see Bar Editor / Event Types]

Coarse changes in a slider's value are made by grabbing and dragging the number bar. Fine changes are made by clicking on either side of the number bar (within the slider's field of travel) to increase or decrease its value.

♦ **Note:** When first entering the Bar Editor, the virtual sliders will appear dormant (having no associated parameters or names). This is because they auto-configure to the current event, and since no current event is chosen, they have nothing to configure to. When there is no current event, the sliders should be ignored.

Grid: and Dur:

These two indicators, found just underneath the virtual sliders, show the current Grid Size and Duration in clocks (1/192 quarter note), as set in the Grid Requester. Since these numbers are used constantly by the software when Adding or Inserting new events or when Quantizing or when **Snap** is on, it's handy to be able to see what they are. These indicators cannot be changed directly, you must change their values from within the Grid requester.

[see Bar Editor / Grid]

Chapter 8 - Event Editor

Overview.	238
Title Bar	241
File Menu	241
Edit menu	242
Options Menu	242
Display Menu	246
Modules Menu	247
The Event List.	247
Event Types	250
Tool List	263
SetSig	263
Signature Display	264
Dup	264
Remove	264
Select	265
Unmark	266
Undo	266
Quiet	266
Scroll	267

Event Editor (continued)

Grid	267
Params.	268
Repeat	268
Event Type:	269
Channel:	269
STEP and BACK	269
Seq:	270
Transport Controls	270
Virtual Sliders.	271

Event Editor

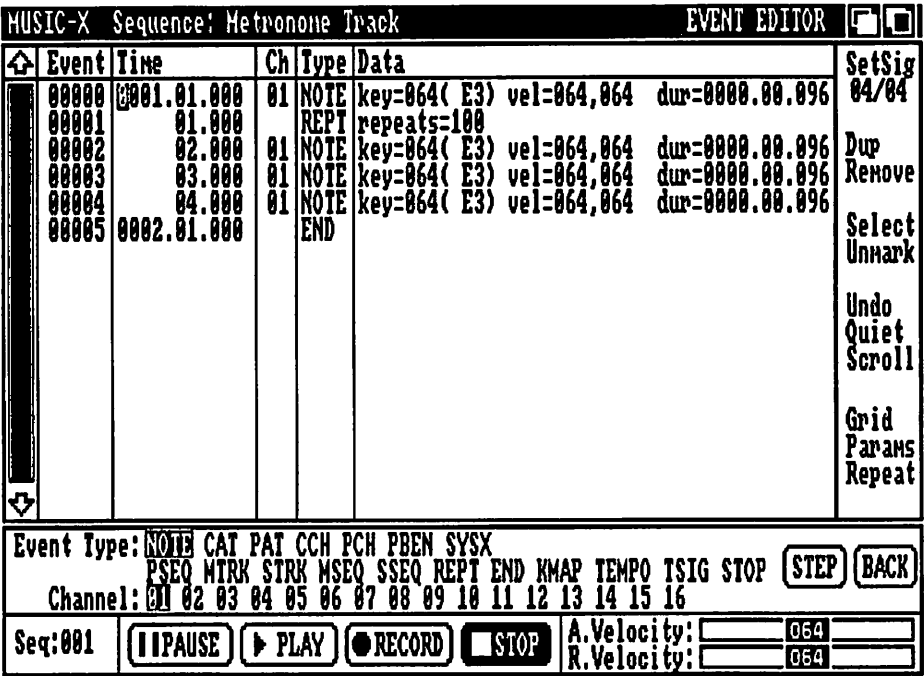


Illustration 8.1 Event Editor Page

Overview

This chapter will describe the features of the Event Editor as they differ from the features of the Bar Editor. Many references are made to terms previously explained in the Bar Editor chapter. For the sake of brevity and to eliminate redundancy, the descriptions in this chapter will assume some familiarity on the part of the reader with the information in the previous chapter. This is probably a safe assumption since the Bar Editor is more “user friendly” and easier to approach as a new user. (Besides, the BarEditor chapter is earlier in the manual!) However, if you are more

comfortable with the technical approach of the Event Editor and have jumped directly to this chapter, then you are probably also the type of person who can pick up the missing pieces.

The Event Editor provides a way of microscopically examining and editing each event in a sequence. Though not as intuitive as its companion the Bar Editor, in some ways it can tell you more about a sequence. The events within the sequence are listed in chronological order. Every event is displayed; nothing is hidden. You see exactly what's in the sequence. Each list entry contains one event, and is divided up into "fields" showing all the parameters of that event. One can scroll through the list and examine it event by event, changing individual parameters of events, or adding and removing events. This particular method of displaying a sequence as a "list" is very similar to what Music-X sees when it reads and plays a sequence.

The Face of the Event Editor

The Event Editor is divided into panels. The largest panel contains the "Event List" which is used to display and edit the events in the sequence. Along the top of the Event List are the names for the various fields in the Event List. Along the right edge of the page is the panel containing the Tool List. Most of these Tool Words are the same as the ones found in the Bar Editor. Below the Event List is a panel containing two quick lists used to edit the event type and MIDI channel parameters of events in the sequence. At the bottom of the page are the Transport Controls, and to the right, the virtual sliders, used to edit various parameters depending on the current event type.

Current Event

As with the Bar Editor, the Event Editor uses the concept of a "current event". All editing functions apply to the current event. The current event is identified by the cursor, which is always visible except when editing groups of Selected events. Moving the cursor through the list changes which event is the current event. The quick lists, and virtual sliders automatically configure themselves to the parameters of the current event.

A Keyboard Approach

Perhaps you prefer typing instead of using the mouse while in the Event Editor. It's possible to edit most fields of most events directly from the Amiga keyboard in word-processor fashion. The only field requiring the use of the mouse is the **Type** field. Click on the quick list labeled **Event Type**, found below the Event List, to change the **Type** field.

Move from event to event by using the Amiga's CursorUp and CursorDown keys. Move through the various parameters of an event with the CursorLeft and CursorRight keys. While the cursor is on a value to be changed, press the plus ("+") and minus ("-") keys on the Amiga's keyboard to increase and decrease the value, or use the numerical keys to type in the new value as you would in a word processor. Changes made to the current event can be undone at anytime before moving the cursor away by pressing the "N" key. ("N" is the keyboard equivalent for the **Undo** command.)

[see Event Editor / Undo]

Keyboard equivalents for the other editing functions are given later as each function is described.

A Mouse Approach

Perhaps you prefer to edit using the mouse, even while in the text-oriented Event Editor. This is supported nicely. The only things you must use the keyboard to change are the start time, stop time, or duration of an event.

Click anywhere in an event line to move the cursor to that line and choose that event as the current event. Then use the quick lists and virtual sliders below the Event List to change the parameters of that event. Changes made to the current event can be undone anytime before moving away from that event by clicking **Undo** in the Tool List.

[see Event Editor / Undo]

Adding and Removing

Adding new events to a sequence from within the Event Editor is accomplished with the Duplicate function. The basic process is to first click the Tool Word **Dup**, or tap the “D” key to insert a copy of the current event into the list. The copied event can then be edited and shaped into a new event.

[see Event Editor / Dup]

Events are removed from the sequence using the Remove function. Each time you click **Remove** or tap the “V” key the current event is taken out of the list and the list is re-sorted to close the gap, leaving the cursor on the next event in the list.

[see Event Editor / Remove]

The rest of this chapter provides descriptions of the individual features of the Event Editor Page.

Title Bar

The Title Bar is displayed along the top of all Music-X pages to show which page is currently being used. In the Event Editor, the name of the sequence currently being edited appears here as well.

The Title Bar becomes the Menu Bar when the right mouse button is pushed and is used to access the menus in the Event Editor.

File Menu

The File menu is the same in both editors and is described near the beginning of the Bar Editor chapter.

[see Bar Editor / File Menu]

Edit Menu

The File menu is the same in both editors and is described near the beginning of the Bar Editor chapter.

[see Bar Editor / Edit Menu]

Options Menu

The Options menu contains the same functions in both editors. However, some of the functions behave slightly differently due to the inherent differences in display methods between the two editors. The following subchapter discusses those differences as they apply to the Event Editor.

Feedback

Feedback is normally heard whenever the cursor is moved to a new event, or whenever one of the parameters of the current event is changed. There are three conditions that will defeat this. 1) If the Event Editor is currently in Select mode. 2) If the **PAUSE** button is on. 3) If **Quiet** is turned on.

[see Event Editor / Quiet]

The **Feedback** item has a pop-out menu with two items. Feedback is sent to MIDI or to the internal Amiga Samples depending on which of these items in the pop-out menu is currently selected (appearing checked).

MIDI

When **MIDI** is checked, all feedback will be played on external MIDI instruments. (A MIDI instrument must be connected for feedback to be heard.)

Amiga

When **Amiga** is checked, all feedback will be played internally on Amiga samples. (There must be samples currently in memory for feedback to be heard.)

[see Amiga Samples chapter]

Output

This item determines to which output bank the current sequence will be sent when played back from the editors or the Sequencer Page. A pop-out menu contains two items. The currently checked item shows the current output bank. These choices are listed below.

[see Sequencer Page / Sequence List / Out]

MIDI

When **MIDI** is checked, the current sequence will be played using external MIDI instruments. (MIDI instruments must be connected for the sequence to be heard.)

Amiga

When **Amiga** is checked, the current sequence will be played using internal Amiga samples. (There must be samples currently in memory for the sequence to be heard.)

Cut

(RightAmiga-“K” from the keyboard)

Cut is used to remove events from the sequence and put them into the Clip.

Choosing **Cut** after predefining a group of events with **Select**, copies those events into the Clip (overwriting its previous contents), then removes them from the sequence. No other events in the sequence are altered.

[see Event Editor / Select]

Choosing **Cut** while only the current event is selected, copies the current event into the the Clip and removes it from the sequence.

Copy

(RightAmiga-“C” from the keyboard)

Copy is used to copy events from the sequence into the Clip. **Copy** does not remove copied events from the sequence; the sequence is left intact.

Choosing **Copy** after selecting a group of events, copies those selected events into the Clip, overwriting the previous contents of the Clip, then deselects them.

Choosing **Copy** while only the current event is selected, copies the current event into the the Clip.

Delete

(RightAmiga-“D” from the keyboard)

Delete is used to remove and discard events from the sequence. **Delete** does not actually involve the Clip.

Choosing **Delete** after Selecting a group of events, removes those events from the score. No other events are altered. The contents of the Clip are not affected. The deleted events are lost.

Choosing **Delete** while only the current event is selected removes and discards the current event. No other events are altered. The contents of the Clip are not affected.

◆ **Note:** If you make a mistake and delete the wrong data by accident, remember that **Recall** can be used to restore the sequence to the condition it was in when last Stored.

[see Bar Editor / File Menu / Recall]

Paste

(RightAmiga-“P” from the keyboard)

Paste is used to place the current contents of the Clip into the sequence at various points. The first event in the Clip is placed at the start time of the current event, and any following events in the Clip are placed in the sequence so that their timing relationship with the first event is the same as it was in the Clip. In other words, the rhythmic shape of the Clip stays the same but is offset as a group when placed at different starting times. The Clip is not erased by this action, so multiple copies of the Clip may be repeatedly pasted into the score. The basic process is to move to the section of the sequence that you want the Clip to be added to, choose the event whose start time will be used when placing the first event in the Clip, and choose **Paste** from the Options Menu.

When **Paste** is used, the contents of the Clip are added to the sequence without affecting any previous events in the sequence. Paste will do nothing if there are any currently Selected events in the sequence. To deselect any events before Pasting, click **Unmark** in the Tool List. Pasting automatically selects all the newly added events. This allows you to further edit the new events after they are added. You may, for instance, change their MIDI channels by clicking on the numbers in the quick list titled **Channel** or even delete them once again with the **Delete** option in this menu. If you are satisfied with the new events as they are, click **Unmark** in the Tool List along the right edge of the Event Editor to deselect the new events and set them into the sequence in their current state

Select All

(RightAmiga-“A” from the keyboard)

Choosing this item selects all events in the sequence, except for the END event. **Select All** can be used in preparation of one of the other group editing procedures such as Copy or Delete.

[see Event Editor / Select]

Show Clip

When **Show Clip** is chosen, the contents of the Clip are shown in the score window instead of the current sequence. This can be used to verify the composition of the Clip before pasting.

While the Clip is being shown in the Sequence List, the Clip may also be played.

To bring the score window back to normal (showing the contents of the Edit Buffer), click once, anywhere within Sequence List.

Display Menu

There is one entry in the Event Editor's Display menu.

Durations

This menu item is used to set the way the end times of events with durations (NOTE, PSEQ, etc.) are displayed in the Event List. When **Durations** is checked, the total duration of each event will be shown at the far right end of the data field. When **Durations** is not checked, the stop time of each event will be shown instead. This can be handy for seeing if events are overlapping, or how much time occurs between the end of one event and the start of another.

Modules Menu

The Modules menu is where access is given to externally loaded modules. There are three modules available from the Event Editor. The menu items that call them are:

Quantize...
Scale Velocity...
Scale Aftertouch...

These modules have been described in detail in the Bar Editor chapter.

[see Bar Editor / Quantizer Module]

[see Bar Editor / Velocity Scaling Module]

[see Bar Editor / Aftertouch Scaling Module]

The Event List

The Event List is the main feature of the Event Editor Page, found just below the Title Bar and occupying most of the screen. This is the work area, where the sequence is displayed and edited. The events are listed chronologically; that is, early events are shown near the top of the list and later events towards the bottom. The Event List scrolls to reveal the changing portions of the sequence being played. New events enter from the bottom and are heard as the sequence plays. Each horizontal line of text in the list describes one event and is divided up into fields to show the different parameters of that event. A cursor is active in the list for keyboard editing, but the mouse may be used just as effectively and probably more quickly. Remember when using the mouse that the current list event is always the one where the cursor is.

◆ **Note:** Auto-scrolling can be turned off and on by clicking **Scroll** along the right edge of the Event Editor Page.

[see Event Editor / Scroll]

Along the left edge of the Event List is a scroll bar. While the sequence is stopped, you can scroll through the list quickly by dragging the slider up and down, or slowly by clicking on the arrow gadgets at either end of the scroll bar and holding the left mouse button. (The list may also be scrolled manually during play if **Scroll** is turned off.)

Along the top of the Event List are the names of the various fields in the list. Following are descriptions of those fields, ordered as found on the screen from left to right.

Event

The **Event** field is used to number the events in the list. This field is used primarily by the software and cannot be edited. The numbers are automatically resorted as events are duplicated or removed. You can use these numbers to get some sense of where you are in the bulk of the sequence.

Time

This field is used to show the start time of each event relative to the beginning of the sequence (not to the beginning of the song; a sequence may arbitrarily start at any point in a song). Time is displayed differently depending on the type of sequence being edited.

In Relative sequences, time is written in Measure, Beat, and Clock format. For example, the time at the beginning of the sequence clock would be written as "0001.01.000" meaning "first measure, first beat, and no clocks". If **Use Zero Origin** has been turned on in the Sequencer Page, then the same clock time would be written as "0000.00.000". To make it easier to see measure boundaries, the measure-number portion of the Time field is only displayed in certain cases: for the first event on the screen, for the first event in a measure, and for the current event. For all the other events, only the beat and clock numbers are shown.

[see Sequencer Page / Options Menu / Use Zero Origin]

In Absolute sequences, time is written in Hour, Minute, Second, and Frame format. Absolute sequences always start at "00:00:00.00". Frames are separated by a period when using Drop Frame and by a colon when using Non-Drop Frame and other frame rates. This is to show that Drop Frame is not an exact frame count.

[see Sequencer Page / Sync Menu / Time Code / Drop Frame]

The start time of any event can be edited by moving the cursor to the **Time** field and either typing in a new number, or pressing the plus ("+") and minus ("-") keys on the Amiga keyboard. When you press the RETURN key or move the cursor from that line, the Event List will be re-sorted in chronological order.

Ch

Read this field title as "Channel".

This field shows the MIDI channel number for events in the list that use MIDI channels. SYSEX events (System Exclusive MIDI messages) and Music-X events (events that control the sequencer internally) don't use MIDI channels. This parameter may be changed using the quick list labeled **Channel**, or, while the cursor is in the **Ch** field, by using the plus ("+") and minus ("-") keys.

[see Event Editor / Channel:]

Type

This field holds the mnemonic (abbreviation) for each event showing what type of event an event is. Possible types are listed on the screen below the Event List in the quick list labeled **Event Type**. The event type of the current event can be changed by clicking on any of the event types provided in the quick list.

[see Event Editor / Event Type]

Data

This field is used to display and edit the parameters for each event, and changes dramatically for different event types. The possible parameters for each event type are described in the following Event Types subchapter.

[see Event Editor / Event Types]

Event Types

This subchapter will explain how each event type is displayed in the Event List, what parameters are available with each event type, and how to edit those parameters.

There are three classes of events in Music-X:

Channel Events: These events send standard MIDI messages as described in the MIDI specification. These are:

<u>Mnemonic</u>	<u>Name</u>
NOTE	(Note)
CAT	(Channel Aftertouch)
PAT	(Polyphonic Aftertouch)
CTL	(Control Change)
PGM	(Program Change)
PBEN	(Pitch Bend)

System Exclusive Events: These events send MIDI messages designed and specified by the instrument manufacturers. All System Exclusive messages fall under the event type:

SYSX	(System Exclusive)
------	--------------------

Music-X Events: These events are used internally by the sequencer in some way. They are:

<u>Mnemonic</u>	<u>Name</u>
END	(End)
SPLI	(Splice)
REPT	(Set Repeats)
STOP	(Stop)
TEMP	(Tempo Change)
TSIG	(Time Signature Change)
KMAP	(Change Keymap)
PSEQ	(Play Sequence)
MSEQ	(Mute Sequence)
SSEQ	(Solo Sequence)
MTRK	(Mute Track)
STRK	(Solo Track)

Common Features

There are certain common features which all event types have and which are edited the same way. These can be explained once and assumed to be consistent.

Start Time

Every event has a Start Time parameter which is changed by moving the cursor to the **Time** field of that event and either typing in the new time using the Amiga keyboard, or pressing the plus (“+”) and minus (“-”) keys to increase and decrease the number under the cursor.

MIDI Channel

The channel numbers of all Channel events may be edited by moving the cursor to the **Ch** field and pressing the Amiga’s plus (“+”) and minus (“-”) keys to increase and decrease the channel number, or by clicking on the numbers in the quick list titled **Channel:**, found below the the Event List. When multiple

events are selected, their channel numbers may be changed simultaneously by using the quick list. The channel numbers of events may be reinterpreted during playback by the Output Channelizer.

[see Sequencer Page / Options Menu / Output Channels...]

Duration

Only certain event types have durations. These event types are:

NOTE	(Note)
TEMP	(Tempo Change)
PSEQ	(Play Sequence)
MSEQ	(Mute Sequence)
SSEQ	(Solo Sequence)
MTRK	(Mute Track)
STRK	(Solo Track)

The Duration parameter of these events is always displayed at the right end of the **Data** field after the word **dur=**. Durations are displayed using the same numbering system as Start Time. That is, as a string of numbers representing Measures, Beats, and Clocks for Relative sequences; and Hours, Minutes, Second, and frames for Absolute sequences. An event with no duration (for instance a TEMP event where the ending tempo should occur on the same clock as the event) appears as all zeros. For example, “dur=0000.00.000” (in Relative sequences), or “dur=00:00:00.00” (in Absolute sequences).

The Durations of events may alternately be indicated using the event's Stop Times instead. (This option is set in the Display Menu.) Stop Times are displayed following the key word **stop=**. This shows the time at which the end of the event occurs (for instance with TEMP events, the stop time shows when the new tempo should be reached).

[see Event Editor / Display Menu]

The Durations of events are edited by moving the cursor over to the **dur=** field and pressing the Amiga's plus ("+") and minus ("-") keys. The Quantizer Module may also be used to alter the Durations of events.

[see Bar Editor / Quantizer Module]

All other parameter types are displayed in the **Data** column as a key word followed by a one or more numeric or alphanumeric fields. The values of these parameters are edited by moving the cursor to the desired field and pressing the Amiga's plus ("+") and minus ("-") keys, or, in most cases, by moving one of the virtual sliders in the lower right corner of the screen.

Following are the event descriptions.

NOTE

(name = Note)

MIDI Key Number: The MIDI key number of a NOTE event follows the word **key=**, and is indicated as a three digit number ranging from 000 to 127, followed by the corresponding musical pitch name and octave number, in parentheses. For example, "key=060(C3)", for middle C. You may also set the current NOTE event to a particular key number by playing that key on the mother keyboard. This last method can be used while in play.

Attack Velocity and Release Velocity: These parameters are displayed following the word **vel=**, as two numbers separated by a comma. Attack Velocity is first. These numbers may also be increased or decreased using the virtual sliders marked **A.Velocity**, and **R.Velocity**. The velocities of NOTE events may be changed with the Velocity Scaling Module as well. Attack Velocities can range from 001 to 127. Release Velocities can range from 000 to 127. The slider also allows a release velocity of 128 to mark a special case. The MIDI specification allows synthesizers to send Note Off messages as Note On messages with attack velocities of zero. This technique makes the MIDI data stream more efficient by eliminating the need to

change the “running status”. Music-X records Note Offs in whichever form it receives them. A Music-X NOTE event will show a Release Velocity of 128 if the Note Off was received as a Note On with zero velocity. This value of 128 would be illegal in MIDI, and is only used as a “flag” by Music-X so it knows to send the Note Off as a Note On during playback. Any other release velocity would cause the Note Off to be sent normally. This parameter is fully editable the same as any other; a Note Off recorded in one form may be changed and played back in the other form.

[see Bar Editor / Velocity Scaling Module]

CAT

(name = Channel Aftertouch)

CAT events send Channel Aftertouch messages to MIDI instruments.

CAT events have three parameters: Start Time, MIDI Channel, and Pressure.

Pressure: This parameter relates to how hard the key is being pressed after reaching the bottom of travel. This value is indicated in the **Data** field of CAT events following the word **press=** as a number ranging from 000 to 127. For example, “press=123”. This number may be increased or decreased by dragging the virtual slider marked **Pressure**. The pressure parameter of CAT events may also be manipulated using the Aftertouch Scaling Module.

[see Bar Editor / Aftertouch Scaling Module]

PAT

(name = Polyphonic Aftertouch)

PAT events send Polyphonic Aftertouch messages to MIDI instruments.

PAT events are like CAT events except that different keys may have different pressure values. Therefore they have an extra parameter for MIDI key number.

PAT events have four parameters: Start Time, MIDI Channel, MIDI key number, and Pressure.

MIDI Key Number: The MIDI key number of a PAT event follows the word **key=**, and is displayed as a three digit number ranging from 000 to 127, followed by the corresponding musical pitch name and octave number, in parentheses. The virtual slider titled **Note** may be dragged to change the MIDI key number.

Pressure: The pressure value is indicated following the key word **press=** as a number ranging from 000 to 127. The virtual slider titled **Pressure** may be used to change the pressure value of a PAT event. The pressure values of a group of PAT events may also be altered with the Aftertouch Scaling Module.

[see Bar Editor / Aftertouch Scaling Module]

CTL

(name = Control Change)

CTL events send Control Change messages to MIDI instruments. A Control Change message causes an instrument to react as if one of the controllers were moved on the instrument itself. Controllers are things like the modulation wheel, breath controller, sustain pedal, etc. Each controller is identified by a number. This family of controllers does not include the pitch bend wheel, which has its own messages, nor does it include the front panel switches used to modify synthesizer parameters making up a program (VCF cutoff, envelope levels, etc.). Synthesizer parameters can be controlled via System Exclusive messages.

CTL events have four parameters: Start Time, MIDI Channel, Controller Number, and Value.

Controller Number: The controller number is displayed following the key word **ctl=**, ranging from 000 to 127.. The controller number for the current CTL event can also be changed by dragging the virtual slider titled **Control**.

Value: The Value parameter represents different things depending on the controller number, but is basically an intensity indicator. To change the value of the current CTL event, drag the virtual slider titled **Value**. The range of possible values is from 000 to 127.

PGM

(name = Program Change)

PGM events have three parameters: Start Time, MIDI Channel, and Program Number.

Program Number: The program number is written following the key word **pgm=**, and ranges from 001 to 128. Program number is also changed by dragging the virtual slider titled **Program**.

PBEN

(name = Pitch Bend)

PBEN events have three parameters: Start Time, MIDI Channel, and Bend Value.

Bend Value: This parameter ranges from -8192 (maximum lowering of pitch), to 0000 (normal pitch), to +8191 (maximum raising of pitch). Bend Value is indicated following the key word **val=**. The bend value for the current PBEN event can also be changed by dragging the virtual slider titled **Bend**.

SYSX

(name = System Exclusive)

SYSX events are used to store MIDI System Exclusive messages as a string of hexadecimal numbers whose meanings are specified by the instrument manufacturers. SYSX events are the only event type with variable length data fields. This is to support variable length MIDI System Exclusive messages. Each SYSX event can

hold up to 6 bytes of data. Longer MIDI System Exclusive messages are stored in Music-X as a series of consecutive SYSX events.

The data portions of SYSX events contain two sets of parentheses. The first set holds the actual data bytes to be sent, and the second set contains a flag that marks the end of the MIDI System Exclusive message.

To change the number of bytes held within a SYSX event, drag the slider titled **Length**. To increase or decrease the value of a data byte within the SYSX event, move the cursor over to the byte to be changed, then press the Amiga's plus ("+") and minus ("-") keys or drag the slider titled **Data**. The data byte is displayed in hexadecimal and the slider is displayed in decimal. This eliminates the need to convert between number bases when needed.

All MIDI System Exclusive messages start with F0 (hexadecimal). This is the "Start of System Exclusive" byte. When Music-X finds a SYSX event in a sequence, it sends out an F0 and then proceeds to send out the series of bytes stored in the SYSX event and in any following consecutive SYSX events. When creating your own SYSX events, it's not necessary to specify the F0 byte.

MIDI System Exclusive messages typically end with F7 (hexadecimal). This is the "End of System Exclusive" (abbreviated as "EOX") byte. Each SYSX event can optionally be tagged with an EOX byte. This will be indicated in the second set of parentheses as (EOX). This causes Music-X to send an F7 byte after sending the data bytes in the SYSX event. In longer MIDI System Exclusive messages, only the last SYSX event will be tagged with an EOX byte. To change the status of the EOX flag, move the cursor to the second set of parentheses and press the Amiga's plus ("+") and minus ("-") keys. When the EOX flag is turned off, the string reads (***).

END

(name = End)

END events have one parameter: Start Time. This is edited normally.

SPLI

(name = Splice)

SPLI events have two parameters: Start Time, and a special message parameter read only in the Event Editor.

Message parameter: One of three messages will be written in the **Data** column of the SPLI event. “IN” means that a punch-in occurred at this time point. “OUT” means that a punch-out occurred at this time point. “LOOP” means that the sequencer rewound the clock at this point while recording with Loop Mode.

REPT

(name = Set Repeats)

The REPT event has two parameters: Start Time, and Repeats.

Repeats: The value of this parameter is indicated following the key word **repeats=** and ranges from 1 to 100. Values 1 through 99 represent the total number of times the sequence will be played. A value of 100 causes the sequence to repeat indefinitely. The repeats value is also changed by dragging the virtual slider titled **Repeats**.

◆ Note: If more than one repeat event is added to a sequence, the last event (preceding the END event) takes precedence.

STOP

(name = Stop)

This event has one parameter: Start Time. This is edited normally.

TEMP

(name = Tempo Change)

TEMP events are Music-X events that cause the sequencer to change the Relative clock rate and therefore the tempo of Relative sequences. Absolute sequences are not affected by TEMP events.

A TEMP event has three parameters: Start Time, New Tempo, and Duration.

Start Time: This parameter determines when in the sequence the tempo will start to change.

New Tempo: This parameter is indicated following the key word **newtempo=**, and ranges from 10 to 300 quarter notes per minute. This value may also be changed by dragging the virtual slider titled **TMP**.

TSIG

(name = Time Signature Change)

TSIG events cause Music-X to change the global time signature that it uses to interpret all Relative sequences during playback.

◆ Note: Future versions of Music-X may allow individual sequences to have their own independent time signatures. For reasons of upward compatibility, it would be a good idea to include TSIG events in all your sequences when TSIG events are used, though they are currently redundant.

TSIG events have three parameters: Start Time, Numerator, and Denominator.

Numerator: This parameter together with the Denominator parameter are displayed as a fraction following the key word **timesig=**. The numerator can range from 1 to 64 but can be no greater than 4 times the denominator. The numerator can also be edited by dragging the virtual slider titled **Beats**.

Denominator: This parameter represents the bottom number in the time signature, showing the type of duration used to make one beat. This number may also be edited by dragging the virtual slider titled **Per**. This parameter ranges from 0 to 4 representing the power of two needed to express each musical note value. (0=whole note, 1=half note, 2=quarter note, 3=eighth note, 4=sixteenth note.) To change the denominator, move the virtual slider.

KMAP

(name = Change Keymap)

KMAP events have three parameters: Start Time, MIDI Channel, and Keymap Number.

MIDI Channel: For KMAP events, MIDI Channel is changed by editing the field following the key word **ch=**. This parameter may also be edited by dragging the virtual slider titled **Channel**.

Keymap number: The Keymap number parameter is indicated following the key word **map=** as a number ranging from 0 to 4. Keymap number may also be edited by dragging the virtual slider titled **Keymap**.

PSEQ

(name = Play Sequence)

PSEQ events have five parameters: Start Time, Sequence, Transposition, Cut, and Duration.

Sequence: This parameter is displayed following the key word **seq=** and ranges from 1 to 250. This parameter may also be changed by dragging the virtual slider titled **Sequence**.

Transposition: This parameter is displayed following the key word **trns=** and

ranges from -64 (64 half steps below), to 000 (no transposition), to +63 (63 half steps above). Transposition may also be edited by dragging the slider titled **Transpose**.

Cut: This parameter is used to determine how the spawned sequence will cut off (stop playing) when the PSEQ event ends. It is displayed following the key word **cut=** as one of three strings. *SELECT USING "+" KEY*

cut= - - - indicates that the spawned sequence is not cut off and will finish playing until its end. Any currently playing PSEQ events nested within the spawned sequence will also be allowed to finish playing with no cut.

cut=Seq indicates that the spawned sequence will stop playing (no new events will be initiated), but any currently playing events within the spawned sequence will be allowed to finish naturally. Any currently playing PSEQ events nested within the spawned sequence will also be allowed to finish playing with no cut.

cut=All indicates that the spawned sequence will stop playing, and any currently playing events within the spawned sequence will be halted abruptly. Any currently playing PSEQ events nested within the spawned sequence will stop initiating new events, but events currently playing will be allowed to finish naturally.

[see Advanced Users / Song Assembly]

MSEQ

(name = Mute Sequence)

MSEQ events have three parameters: Start Time, Sequence, and Duration.

Sequence: This parameter is displayed following the key word **seq=** and ranges from 1 to 250. This parameter may also be changed by dragging the virtual slider titled **Sequence**.

SSEQ

(name = Solo Sequence)

SSEQ events have three parameters: Start Time, Sequence, and Duration.

Sequence: This parameter is displayed following the key word **seq=** and ranges from 1 to 250. This parameter may also be changed by dragging the virtual slider titled **Sequence**.

MTRK

(name = Mute Track)

MTRK events have three parameters: Start Time, Track, and Duration.

Track: This parameter is displayed following the key word **trk=** and ranges from 1 to 20. This parameter may also be changed by dragging the virtual slider titled **Track**.

STRK

(name = Solo Track)

STRK events have three parameters: Start Time, Track, and Duration.

Track: This parameter is displayed following the key word **trk=** and ranges from 1 to 20. This parameter may also be changed by dragging the virtual slider titled **Track**.

Tool List

Along the right edge of the Event Editor Page is a list of tool words used during editing. They offer the most often used editing functions in the Event Editor and are provided next to the Event List for easy access. Some are mode switches, some open windows where parameters can be tweaked, and some are direct commands to Music-X that cause things to happen instantly.

Tool words are active when highlighted in white, and inactive when written in blue. Tool words are activated, deactivated, or used by clicking them.

The following subchapters give an explanation of each tool word and any windows or parameters associated with that word.

SetSig

(" / " from the keyboard)

Read this tool word as "Set Signature". **SetSig** is used when editing sequences containing Time Signature Change events to adjust the **Time** fields of events to the current time signature.

Normally, the **Time** fields of events are readjusted automatically while the sequence plays to conform with Time Signature Change events as they are encountered. When the sequence is stopped, the **Time** fields stay set in the last valid time signature. If you then manually scroll to a portion of the sequence with a different time signature, the **Time** fields may appear incorrect. Playing from that point would resize the lines, but if you don't want to play the sequence, you can use **SetSig** to manually adjust the **Time** fields to the new time signature.

If there are multiple Time Signature Change events currently displayed in the Event List, **SetSig** uses the last valid Time Signature Change event at the current cursor position as a reference.

Signature display

Under the SetSig tool word is a display of the current time signature. It displays the current time signature, as set by either the global time signature or the last TSIG event encountered while playing a sequence from within the editor. This is purely an indicator and cannot be edited. The time signature display is always highlighted in white to show that it's active.

◆ Note: The global time signature can be changed from the Sequencer Page, or from the **Params** window in either Event Editor, or by inserting a TSIG event into the sequence.

[see Sequencer Page / Time Signature Window]
[see Bar Editor / Params / Time Signature]

Dup

("D" from the keyboard)

Duplicate is used to add new events to the sequence. Each time you click **Dup**, a copy of the current event is inserted into the list. After duplicating an event, edit the copy (change its event type, its start time, etc.) to reshape it into a new event.

Remove

("V" from the keyboard)

Remove is used to remove events from the sequence, one at a time. Clicking this tool word removes the current event from the list. The list is re-sorted and the cursor moves to the next event in the list. The tool word **Remove** stays highlighted briefly as the current event is removed, then returns to normal. END events cannot be removed.

If you remove an event accidentally, you can restore it by pressing the **HELP** key on the Amiga keyboard. Repeatedly pressing the **HELP** key will restore up to the last 16 events previously removed from the sequence.

◆ **Note:** Moving between editors resets the **HELP** key. Restore accidentally removed events before switching editors.

Select

("E" from the keyboard)

Click on this word to activate Select mode. While **Select** is highlighted, clicking on any event in the list will select that event for future operations. By selecting multiple events, one can do group operations easily. The event numbers of selected events are highlighted in blue to show that they are selected. Clicking again on any selected event will deselect that event. All selected events may also be deselected by clicking **Unmark** (see below). When one or more events are selected, the cursor disappears. Selecting must be done with the mouse.

Once an event or group of events is selected, many other editing functions can be used to edit them collectively. For instance, clicking on a new event type in the **Event Type** list will change all the currently selected events to the new type. Clicking on the **Channel** quick list will change their MIDI channel assignments. The Cut and Paste operations in the Options menu are designed to work on groups of selected events. The Quantizer, Velocity Scaler, and Aftertouch Scaler modules can all be used to operate on selected events.

To turn off Select mode, click **Select** again.

[see Event Editor / Modules Menu]

[see Event Editor / Options Menu]

Unmark

("U" from the keyboard)

Clicking **Unmark** causes all currently selected events to become deselected. The word **Unmark** stays highlighted briefly while the events are being unmarked, then returns to normal. **Unmark** can be used at any time.

Undo

("N" from the keyboard)

Undo is used when editing an event to restore the event to its original condition. This may be used at any time before moving the cursor away from the event line. For example, change the attack and release velocities of a Note event, then click **Undo**; the velocities are returned to normal.

◆ Note: Editing a note event's MIDI key number by pressing a key on the mother keyboard fixes the current condition of the event and cannot be undone.

Quiet

("Q" from the keyboard)

Quiet is an optional mode switch, and may be used with any other mode. **Quiet** turns off the audible feedback heard when Note events are edited in the Event Editor. Click **Quiet** to turn it on and off. **Quiet** does not affect the play-back of events when the editor is in play or record.

[see Event Editor / Options Menu / Feedback]

Scroll

("L" from the keyboard)

Scroll is an optional mode switch, and may be used at any time. **Scroll** controls the auto-scroll function of the graphic score window. Click **Scroll** to turn it on and off.

If **Scroll** is on (highlighted), then when the editor is in play or record, the Event List will automatically scroll along through the sequence in time. New events enter from the bottom of the list as they are played.

If **Scroll** is off, the score window does not automatically scroll as the sequence is played, but may still be manually scrolled.

Grid

("G" from the keyboard)

This tool word is used to call the Grid requester where the current **Grid Size** and **Duration** parameters may be altered. This tool word may be used at anytime including during play or record.

The two parameters Grid Size and Duration are used by other functions in the Editors. Grid Size and Duration are both used by the Quantizer Module. Grid Size is used by the **STEP** button when recording in step mode.

The features of the Grid requester are discussed fully in the Bar Editor chapter.

[see Bar Editor / Grid]

Params

(SHIFT-“P” from the keyboard)

This tool word is used to open the **PLAY PARAMETERS** window where various parameters governing the way the sequence is played may be altered. Some of these parameters are also found in the Sequencer Page but have been made available here for convenience.

Descriptions of the various parameters available from this window are available in the Bar Editor chapter.

[see Bar Editor / Params]

Repeat

(SHIFT-“R” from the keyboard)

The last of the tool words in the Tool List is **Repeat**, found directly below the word **Params**. **Repeat** is an optional mode switch, and may be used at any time. **Repeat** controls the auto-repeat function of the graphic score window. Click **Repeat** to turn it on and off.

This toolword is turned on (highlighted in white) by default. When on, it causes the sequence to auto-repeat *while in the editor*. When the END event is reached during play, the clock returns to the beginning of the sequence and starts again.

◆ Note: When Playing In Context, the auto-repeat function is defeated to give true playback.

[see Bar Editor / Params / Play In Context]

Event Type:

Below the Event List window is a quick list labeled **Event Type**. This lists all the possible event types that may be edited from the Event Editor (only SPLI events may not be edited). The list is divided into two rows. The top row contains MIDI event types and the bottom row contains Music-X event types. By clicking on the controls listed here, one changes the event type of the current event, or of a group of selected events. The quick list is provided on the screen to speed up the editing process. The event types are discussed in detail in the Event Types subchapter.

[see Event Editor / Event Types]

[see Event Editor / Select]

Channel:

Below the Event Type list is another quick list of the 16 possible MIDI channels. This quick list is provided on the screen to facilitate quick mouse editing of the MIDI channel field of any event. After moving the cursor to the event to be changed, click on the row of channel numbers. The event's MIDI channel assignment will change accordingly. To change the MIDI channel of an event, using the keyboard, move the cursor to the **Ch:** field and use the Amiga's plus ("+") and minus ("-") keys to increase or decrease the channel number.

STEP and BACK

("]" from the keyboard)

("[" from the keyboard)

To the right of the quick lists are two buttons labeled **STEP** and **BACK**. These are used during Step recording to advance or rewind the clock. The editor must be in record mode, with the **PAUSE** button on. These buttons function identically with their counterparts in the Bar Editor.

[see Bar Editor / Params / Play In Context]

Seq:

Read this indicator as “Sequence Number”.

The **Seq:** indicator is found in the lower left corner of the Event Editor Page. It’s used to indicate which sequence is currently being edited. The current sequence can be changed in two ways. First, by bringing a new sequence into the editor, using **Select...**, in the File menu. Second, by Storing the current sequence to somewhere besides its original location, using **Store...**, in the File menu.

Transport Controls

The transport controls in the Event Editor work similarly to the transport controls in the Bar Editor with these differences:



Illustration 8.3 Event Editor Transport Controls

Clicking **PLAY** causes the sequence to be played starting with the first event visible at the top of the list. If **Scroll** is on, the cursor moves through the list as the sequence plays, showing which event is currently being played. When the cursor reaches the bottom of the screen, the list begins to scroll. New events enter from the bottom and are played as they are shown.

Clicking **RECORD** first clears the list and then begins to build the list as new events are recorded. If **Scroll** is on, the list builds downward towards the bottom of the screen with the cursor on the latest event. When the cursor reaches the bottom of the screen, old events are pushed off the top as new events are recorded.

Virtual Sliders

In the bottom right corner of the Event Editor Page are two virtual sliders. These sliders have a number bar where most sliders have a drag bar. This means they can be used to both indicate a value, and change that value. These sliders are provided to give quick access to some of the parameters in the current event. Each time a new event is grabbed, the sliders reconfigure themselves to the parameters of that event. Their names and ranges change automatically depending on the current event type. The basic process when editing events is to first click on an event, then adjust the sliders.



Illustration 8.4 Virtual Sliders

Coarse changes in a slider's value are made by grabbing and dragging the number bar. Fine changes are made by clicking on either side of the number bar (within the slider's field of travel) and holding the mouse button, to increase or decrease its value.

◆ **Note:** Detailed descriptions of the possible configurations of these sliders are given in the descriptions of the event types.

[see Event Editor / Event types]

Chapter 9 - Filters Page

Overview.	273
Title Bar.	278
Mode Menu.	278
File Menu	278
Modules Menu.	280
Channel.	280
Remap Panel.	281
MIDI Message Panel.	285
Keyboard Map.	287
Data Echo.	288

Filters Page

MUSIC-X File: Untitled FILTERS

Channel: **1** 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Event Type	Remap	Filter
Note	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	ENABLE
Channel Aftertouch	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	100
Poly Aftertouch	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	100
Program Change	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	ENABLE
Control Change	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	ENABLE
Pitch Bend	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	100
SET ALL	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	

PORTAMENTO ON PORTAMENTO OFF

VIBRATO ON VIBRATO OFF

ALL NOTES OFF RESET CONTROLS

OMNI ON OMNI OFF

POLY MODE MONO MODE

LOCAL ON LOCAL OFF

Keyboard Map:

OFF 1 2 3 4

Data Echo:

RE-OUT INTERNAL

Illustration 9.1 Filters Page

Overview

The Filters Page is reached from any of the four main Music-X Pages, including the Sequencer Page, by choosing **Set Filters** in the Mode menu or by holding the RightAmiga key and pressing the “F” key.

The Filters Page is used to process incoming MIDI messages before they are used by the rest of Music-X. For instance, when recording, the Filters Page is patched into the data path from the mother keyboard to the Sequencer Page. Incoming

messages first pass through the Filters Page, where they are processed, and then on to the Sequencer Page where they are recorded.

There are five basic processes in the Filters Page. The Filters Page can—

Transmute note messages into other event types

Obstruct the passage of chosen message types

“Thin out” the quantity of chosen message types

Change the MIDI channel of chosen message types

Echo messages to the internal voices or back to MIDI OUT

These five processes constitute the operation of one filter. There are 16 complete filters, one for each channel. The 16 filters are constantly active as long as Music-X is running, no matter what page you are in. The only exception is during actual data transfer in the Librarian Page. Most of the time only one filter will be needed. However, the Filter Page’s real strength lies in interpreting multiple MIDI channels in real time, as when recording live data from a band of MIDI instrumentalists through a merge box. Each input channel can have its own unique filter setup.

Transmutation of Note events into other event types is accomplished using Keyboard Maps. A Keyboard Map allows every key on the input keyboard to be interpreted differently. A keymap is basically a “look up table” relating the 128 possible key numbers to a set of possible resulting events. One particular event is specified for each note in the table. Each time a note event is passed through the keymap, the specified event for that key number is substituted in its place. By changing the table, different map configurations are created. Keymap tables are edited with the Keymap Editor module.

Since a keymap bases its effect on key number, rather than on channel number, any channel may be passed through it with the same effect. It's therefore possible to use the same keymap with different filters. In practice, the four keymaps are shared by the 16 filters. When a filter is routed through a keymap, Note events on that channel bypass the normal filtering path and are reinterpreted by the keymap instead.

[see Keymap Editor chapter]

The face of the Filters Page

The Filters Page has four main panels. The first panel, at the top of the page contains a row of channel buttons, and is used to set the "current filter" being edited by the rest of the controls in the Filters Page.

The second panel contains the actual filter controls used to thin out or ignore chosen event types, and rechannelize them.

The third panel is the MIDI Message Panel, which contains a set of buttons used to send MIDI "mode messages" to any devices which may be listening on the current channel. (The current channel number is the same as the current filter number, and is set in the top panel.) Mode messages can be used to configure or initialize the synthesizers in your setup by remote control. The buttons in the Remap panel are not actually filter controls; they do not affect the current filter settings in the other panels. However, they have been included in the Filters Page because they too are used to configure the overall MIDI setup.

The fourth panel contains two sets of switches. On the left, are the Keyboard Map switches that, if turned on, cause Note events on the current channel to be routed through one of the four keymaps instead of through the normal Filters Page controls in the second panel. On the right, are the Data Echo switches. These determine whether data on the current channel, after being modified by the keymaps or other controls, will be echoed back out the MIDI port, or played on the internal Amiga samples, or not echoed at all.

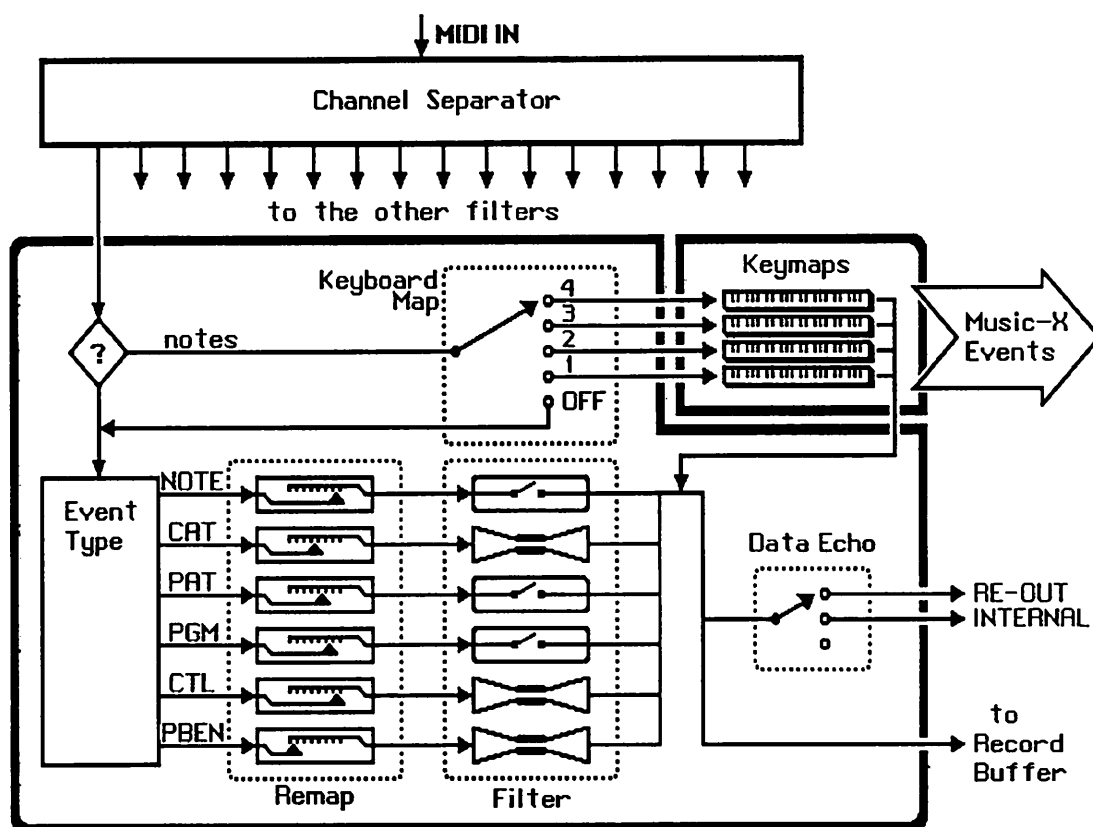


Illustration 9.2 Music-X Filtering Process Schematic

The above illustration gives a flow diagram for the filtering process. Let's follow the data path for incoming MIDI events, as they pass through the Filters Page in a typical setup before being recorded

- ❑ Incoming events are first separated by **Channel** into the 16 filter slots. (The following steps are described for one slot.)
- ❑ If a **Keyboard Map** is switched on for the current slot, Note events are skimmed off and sent to the specified map. Otherwise, if keymaps are

switched off (as in the illustration), note events continue on with the rest of the events.

- ❑ Events on the current channel are further separated by **Event Type** into the six groups (Notes, Channel Aftertouch, etc.).
- ❑ Each Event Type group then passes through its own channel remapper and is given a new channel number according to the current **Remap** setting.
- ❑ Each Event Type group then passes through its own gate in the **Filter** section. Some gates are simple enable or disable switches that can filter out all the events of one type. Others are selective. In the illustration, the hourglass shaped filters represent selective gates that can “squeeze down” the quantity of events so that only a percentage of them pass through.
- ❑ At this point, all Event Types are pooled together, and are joined by events returning from Keymap processing. (Note events transmuted into Music-X events and Music-X commands by the keymaps are sent directly to the sequencer and are not pooled with normal events. This is shown in the illustration by the large arrow titled “Music-X Events”.) The pooled events for the current slot are then optionally echoed to MIDI OUT or to the internal voices, if one of the Data Echo switches (**RE-OUT**, or **INTERNAL**) is turned on for this slot.
- ❑ The events then pass from the Filters Page and on to the rest of Music-X. If the sequencer is currently in record mode (as assumed for the illustration), they are captured in the Record Buffer.

The rest of this chapter explains the individual features of the Filters Page.

Title Bar

The Title Bar always shows the program name at the left, and the page name at the right. In the Filters Page, the name of the last loaded or saved filter file will appear here as well. If no filters have been loaded or saved then the current filter will be untitled, i.e., "File: Untitled".

The Title Bar becomes the Menu Bar when the right mouse button is pressed, showing the titles of available menus. To access the menus in the Filters Page, press the right mouse button and point to the desired menu title in the Menu Bar. The chosen menu will "pull down", listing the available menu items. To choose a menu item, point to it, and release the right mouse button while the item is highlighted.

Mode Menu

The Mode menu is available from each of the four main pages in Music-X, including the Filters Page, and is used to move between them. This menu has been described in the Sequencer Page chapter.

[see Sequencer Page / Mode Menu]

File Menu

The File menu is used to save or load the settings for the current filter. It also contains a generalized file-initialization function that "zeroes out" all 16 filters to the same basic starting configuration.

Load Filter...

This menu item is used to load a previously saved filter file from disk into the current filter slot.

Before loading the file, choose the target filter to be overwritten by the load. (Click on the **Channel** button for the chosen filter.) After selecting the current filter, choose the **Load Filter...** menu item to call the file requester under the current guise of “Load Filter”. If you decide not to load the file, click **CANCEL** to exit the file requester safely. Otherwise, use the file requester to locate the drive and drawer that the filter file is in. Then click **OK** to load the file. The filter file will be loaded into the current filter, overwriting the previous contents.

[see File Requester chapter]

Save Filter...

This menu item is used to save the settings for the current filter into a filter file on disk. No other filters are saved.

◆ Note: All 16 filters can be saved in a performance file from the Sequencer Page.

[see Sequencer Page / File Menu / Save Performance]

Before saving the the file, choose the filter whose settings you wish to save. (Click on the **Channel** button for the chosen filter.) After selecting the current filter, choose the **Save Filter...** menu item to call the file requester. The file requester will be titled “Save Filter”. If you decide not to save the file, click **CANCEL** to exit the file requester safely. Otherwise, use the file requester to locate the drive and drawer that the filter file should be saved to. Then use the file requester to name your new filter file. The recommended file extension for filter files is “.Filter”. If you are resaving a previous file, use the same name to overwrite the old file with the new file. Then click **OK** to save the filter file.

[see File Requester chapter]

Initialize All

The **Initialize All** item is used to reset all 16 filters to their basic default settings. That is, with each incoming channel being remapped to the same internal channel, and with all event types enabled. Keymaps and RE-OUT are turned off in all cases. This can be used as a starting point when creating your own filter configurations.

To initialize the Filters Page, choose **Initialize All** from the File menu.

Modules Menu

As in other pages, the Modules menu is used to access external programs designed to work with Music-X. Music-X comes with one Module installed in the Filters Page. This Module is the Keymap Editor. The Keymap Editor is called by choosing the following menu item.

Edit Keymaps...

The Keymap Editor is used to edit the four keymaps used by the Filters Page to transmute incoming note events into other types of events. This allows for even more convoluted MIDI remapping tricks like splitting and transposing a non-splitting keyboard, or triggering sequences from the keyboard.

Choosing this menu item loads the Keymap Editor into memory from disk and then moves you into it.

[see Keymap Editor chapter]

Channel:

The top panel in the Filters Page contains a row of buttons, labeled **Channel:**, numbered 1 through 16, representing the 16 MIDI channels, and the 16 filters. These channel buttons have two functions in the Filters Page.

The first function is to determine which filter is currently being edited by the Filters Page. Only one filter can be edited at a time. This is called the “current filter”. By changing the current filter, then editing the controls in the Filters Page, one can edit the 16 filters.

The second function is to determine the output channel that messages from the MIDI message Panel will be sent on.

Change the current filter, and the current channel, by clicking on any one of the Channel buttons in the Channel Panel.

Remap Panel

The second panel from the top of the filters page is the Remap Panel. The controls in this panel are used to edit the filter settings for the current input channel (as set at the top of the page with the **Channel** buttons). The remap panel is divided into three sections labeled: **Event Type**, **Remap**, and **Filter**. Explanations of these follow.

Event Type	Remap																Filter	
Note	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	ENABLE	
Channel Aftertouch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		100
Poly Aftertouch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		100
Program Change	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	ENABLE	
Control Change	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	ENABLE	
Pitch Bend	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		100
SET ALL	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		

Illustration 9.3 Remap Panel

Event Type

This column lists the six incoming MIDI event types that can be rechannelized. The choices are:

Notes
Channel Aftertouch
Poly Aftertouch
Program Change
Control Change
Pitch Bend

Each row of controls in the Remap panel corresponds with the event type shown in the column at the left end of the row. There are six rows, giving one set of controls for each event type. The Event Type names are for display only and are not controls themselves. Event Types are explained fully in the Bar Editor chapter.

[see Bar Editor / Event Types]

Remap

Extending to the right of each event type is a row of numbers from 1 to 16. Above them is the title **Remap**. This is where the original channel numbers for the events are “remapped” into new channel numbers. One of the numbers will be highlighted in white for each event type. This number shows the new channel number which will be attached to events of that type as they are remapped. The Remap numbers are changed by clicking on them.

By setting different Remap channels for each event type it's possible to take the data being sent by the mother keyboard on one MIDI channel and selectively “scatter” it out to multiple MIDI channels. It would be possible, for instance, to send Note events to a piano module, while using Aftertouch to bring in a string pad that the sequencer is playing. Or, to send Program Changes to a sampler while Pitch Bending a drum machine.

At the bottom of the Remap section, is a row of numbers titled **SET ALL**. Clicking on one of these numbers causes all six event types to be remapped to the same channel.

Example: Using the Remap panel to record parts for instruments on different MIDI channels with a mother keyboard that only sends out on one MIDI channel (as with the classic DX7).

- ☐ Select channel 1 in the **Channel** panel at the top of the page. This is the MIDI channel our mother keyboard will be transmitting on.
- ☐ Let's say our bass part will be on channel 2. Click on the number "2" in the **SET ALL** row of the Remap panel. The Remap numbers for the six event types should all now show channel 2 as being selected. (The number "2" should appear highlighted in each row.) Notes transmitted on channel 1 from the mother keyboard will now be remapped to channel 2.
- ☐ Turn on the Data Echo switch titled **RE-OUT** to enable notes played on the mother keyboard to be sent to the bass module (that we are assuming is set to receive on channel 2).
- ☐ Go to the Sequencer page and record the bass part. When done, return to the Filters page.
- ☐ Let's put a string part on channel 3. Click on the number "3" in the **SET ALL** row of the Remap panel. Notes transmitted on channel 1 from the mother keyboard will now be remapped to channel 3.
- ☐ Go to the Sequencer page and record the string part.

Filter

At the right end of the Remap panel is a column of controls labeled **Filter**. These controls can be used to filter out any of the six event types on the current channel. There are two types of controls. Switches of the first type, the **ENABLE** switches, have two states. They either enable or disable the passage of events. Switches of the second type, the sliders, can be adjusted to enable the passage of a percentage of the events. Short discussions of the two types follow.

ENABLE Switches

ENABLE switches are used for Note, Program Change, and Control Change events. With these events you either want all of them or none of them. If you want any Notes, you want all of them. If Program changes are on, you want all of them. Certain control changes might be nice to scale down (breath controller, for instance) but there are many types of controllers. You never know what else might be eliminated if you start thinning out the controller stream. For instance, you wouldn't want to miss any Sustain Pedal presses. For this reason, Control Change messages have been given an **ENABLE** switch.

An **ENABLE** switch is turned on and off by clicking on it. When it's highlighted in white, the event type is enabled. When it appears darkened, the event type is disabled.

Percentage Sliders

Certain synthesizer controllers are notorious for spitting out large amounts of MIDI data when used. Especially Pitch Bend and Aftertouch. These controllers can eat up memory space quickly when recording. One solution is to turn them off when recording; to disable all events of that type. In most cases, this is fine. However, when recording a musical part where they are needed, the percentage sliders can help by cutting down the number of controller messages you record without having to resort to turning them off completely. This is done by ignoring consecutively received events of the same type whose values are identical or very similar. In other words, if the change is too small to notice, why

record it? You save memory, but still get the overall effect. The range of these sliders is from 000 percent to 100 percent. If the slider is at 100 then all the incoming data is accepted. If the slider is at 000, no data is accepted. The middle ranges of the slider are used to set the allowable difference between consecutive events. The smaller the slider value, the greater the difference allowed between the values of consecutive events, and the less events will pass through. The sliders can be adjusted by dragging them, or by clicking on either side of the drag bar in the center.

MIDI Message Panel

The third panel in the Filters Page contains a set of buttons that are used to send specific MIDI control codes to any synths on the currently chosen MIDI channel (selected at the top of the Filters Page). Clicking on one of the buttons sends the message associated with that button. These buttons can be used to set the receive modes (Omni, Mono, etc.) on the synths in your system, by remote. These buttons are handy when using older synths because they prevent you from having to remember how to set them from their front panels (some used obscure button combinations to change modes).

[See Event Editor / Event Types / SYSX]

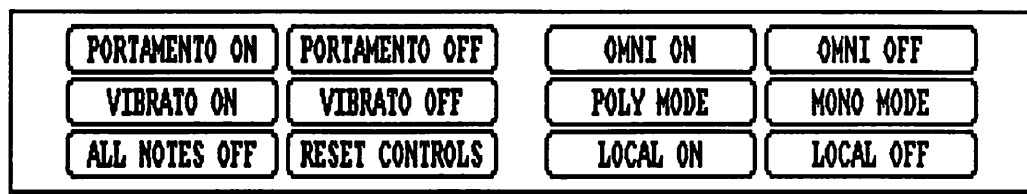


Illustration 9.4 MIDI Message Panel

Following are descriptions of the message that each button sends, and what the result is. Some early MIDI synths may not have correctly implemented the MIDI specification for reception of control messages. Your MIDI synthesizer or device must have these functions implemented in its software for the buttons to have the correct effect. The message strings sent by the MIDI Message Panel are provided here for comparison with your synthesizer's specifications. The messages are shown written in hexadecimal notation as well as in decimal. Numbers written in hexadecimal are followed by the letter "H". All messages are three bytes long, starting with the hexadecimal number BnH (176 plus the current MIDI Channel number (n) minus 1) (Controller Status byte), then the controller number, and the value for that controller.

◆ Note: It is also possible to imbed these MIDI control codes in sequences. Use a Control Change event, with the Controller Number parameter set to the first decimal number in the example, and the Value parameter set to the second decimal number.

Portamento On 41H, 7FH (65,127)
(Sliding between pitches)

Portamento Off 41H, 00H (65,0)
(No sliding)

Vibrato On 01H, 7FH (1,127)
(Wavering of pitch)

Vibrato Off 01H, 00H (1,0)
(No wavering)

Omni On 7DH, 00H (125,0)
(Play notes received on all channels)

Omni Off 7CH, 00H (124,0)
(Play notes received on one channel)

Poly Mode 7FH, 00H (127,0)
(Play more than one note at a time)

Mono Mode 7EH, 00H (126,0)
(Play only one note at a time)

Local Control On 7AH, 7FH (122,127)
(Sound notes played on synth's own keyboard)

Local Control Off 7AH, 00H (122,0)
(Ignore notes played on synth's own keyboard)

All Notes Off 7BH, 00H (123,0)
(Release hanging notes)

Reset Controllers 79H, 00H (121,0)
(Set the value of all controllers to their default positions.
Example: Center the pitch bend wheel, C etc.)

Keyboard Map

At the bottom left of the Filters Page are five switches labeled **OFF**, **1**, **2**, **3** and **4**. These switches are used to turn on one of the four keyboard maps for the current filter (as selected at the top of the page).



Illustration 9.5 Keyboard Map

A keymap allows you to reinterpret every incoming MIDI note on the current channel differently. There are four keymaps. Only one map can be applied to each incoming MIDI channel at a time but a map may be used by more than one channel. The four keymaps may be edited from the Keymap Editor Page (accessed via the Modules menu).

[see Filters Page / Modules Menu / Edit Keymaps]

[see Keymap Editor chapter]

When a keymap is switched on for a channel, then Note events bypass the current Remap panel settings in favor of the keymap. They are still affected by the Data Echo buttons, however. The other five event types are still interpreted normally, using the Remap panel settings.

Data Echo:

In the bottom right corner of the Filters Page are two buttons labeled **Data Echo:**. This determines if the data received on the current channel will be echoed on the MIDI OUT port (to play external synthesizers) or will be directed to the Amiga's own internal sampled voices. The Data Echo buttons are turned on and off by clicking on them. They appear highlighted in yellow when on. Only one may be active at once. Turning one on turns the other off. To turn both off, click on the currently highlighted button.

RE-OUT

When this switch is on, data received on the MIDI channel specified at the top of the page will be echoed back out over MIDI after being sent through any filters and active keymaps. **RE-OUT** is similar to MIDI THRU, except that data is sent out the MIDI OUT port, after being modified. This can be used when recording parts for sound modules while playing the mother keyboard. When this switch is turned on, the **INTERNAL** switch is automatically turned off.

Example: Recording drums on MIDI channel 10 from the mother keyboard.

- ☐ Set the drums to receive on channel 10. Set the mother keyboard to send on 10 (if it can't then remap it to 10 as explained before), and turn its volume down (or use its "local off" setting).
- ☐ Set the current Channel in the Filters Page to the channel that the mother keyboard is transmitting on.
- ☐ Turn on the **RE-OUT** switch for the current channel. This enables notes played on the mother keyboard to be sent to the drum machine at the same time as they are recorded.
- ☐ Click on the number "10" in the **SET ALL** row of the Remap panel to assure that all event types coming from the mother keyboard are remapped to channel 10.
- ☐ Set the **ENABLE** switches and percentage sliders to allow the passing of all messages on the current channel (except for maybe Aftertouch, since drum machines don't typically use these messages).
- ☐ Make sure the **Keyboard Map** switches are set to **OFF**.
- ☐ Playing the mother keyboard will now trigger the drum module.
- ☐ Go to the Sequencer Page and record the drum part manually by playing the mother keyboard.

INTERNAL

When this Data Echo switch is on, data received over the current MIDI channel will be redirected to the internal Amiga samples after being passed through any filters and active keymaps. Think of the sampled voices as another bank of 16 MIDI channels, (with the limit of four voices sounding at one time). When **INTERNAL** is

turned on, **RE-OUT** is automatically turned off. With this button, it's possible to play Amiga samples from the mother keyboard.

Example: Using the mother keyboard to record the music for a computer game your friend is writing.

- ☐ Load up the samples you'll be using in the game.
- ☐ Set the current channel in the Filters Page to the channel that the mother keyboard will be sending on.
- ☐ Set the Remap panel to the channel that the first sample part to be recorded is on. (Click the number in the **SET ALL** row that corresponds to the sample's position in the Samples Page list.)
- ☐ Turn on the **INTERNAL** switch.
- ☐ Notes played on the mother keyboard will now trigger the internal Amiga Samples. (After recording a sequence for the internal voices, remember to set its Output Bank field to **Int.**)

[see Amiga Samples Page]

[see Sequencer Page / Sequence List / Out]

If both of the **Data Echo** buttons are turned off, data can still be recorded, but is not echoed. This might be used when recording the part to be played back on the mother keyboard, when that keyboard doesn't have "local off" capability. In other words, since the mother keyboard is already sounding the notes played on its keyboard during record, we don't need to send them to it again.

Example: Recording without Data Echo.

- ☐ Set the mother keyboard to send and receive on MIDI Channel 1.
- ☐ Set the current channel in the Filters Page to channel 1 by clicking the “1” button in the **Channel** panel.
- ☐ Click on the number “1” in the **Set All** row of the Remap panel.
- ☐ Turn off both **Data Echo:** buttons to prevent the notes from the mother keyboard from being echoed back to the mother keyboard during record, causing doubled notes.
- ☐ Go to the Sequencer Page, turn up the volume of the mother keyboard so you can hear what you play, and record the keyboard part.
- ☐ When the sequence is played back, it will be played on the mother keyboard (assuming the Output Channelizer is not active.)

[see Sequencer Page / Options Menu / Output Channels...]

Chapter 10 - Keymap Editor

Overview. 293

Title Bar. 296

File Menu 296

Mode Panel 298

Miniature Keyboard 302

Action List. 308

Parameter Window 314

MIDI Recording 314

Keymap Editor

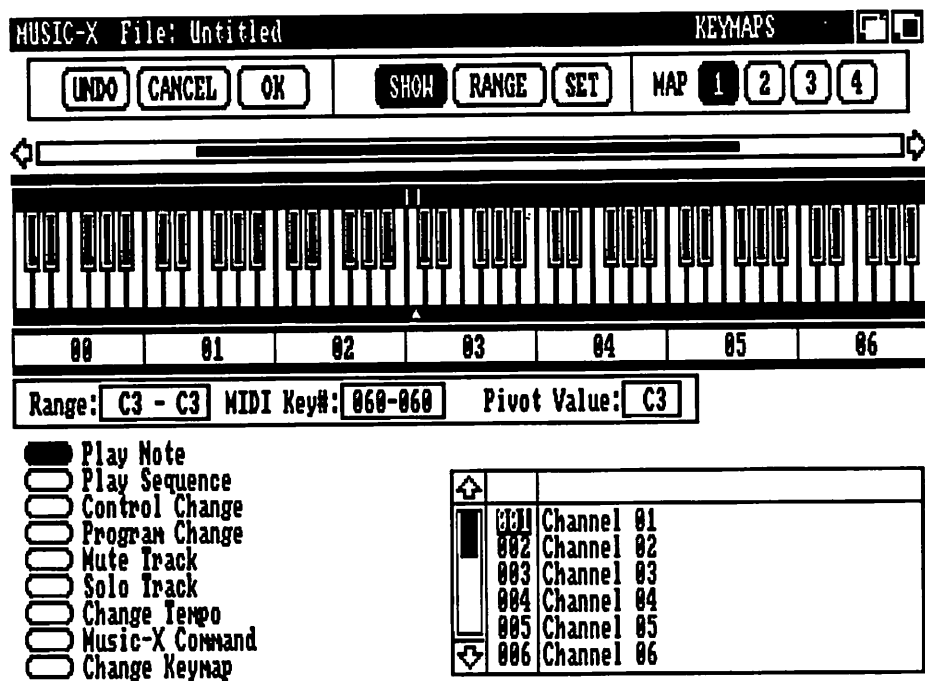


Illustration 10.1 Keymap Editor Page

Overview

The Keymap Editor Page is accessed by choosing **Edit Keymaps** from the Modules menu of the Filters Page. Think of it as an extension of the Filters Page. It's used to edit the four Keyboard Maps used by the Filters Page to transmute or reinterpret incoming Note events.

What is a Keyboard Map?

Imagine a row of keys in a silent keyboard. Now imagine another row of keys with strings attached to the first keys. Every time you press one of the first keys, a string gets pulled, and the same key on the second row gets played. Now imagine what would happen if we rearranged the strings so that they were out of order. We could play any key with any other key. This is the idea behind remapping a keyboard. A keymap allows you to rearrange the notes of the mother keyboard anyway you want, in software. Imagine again that those strings can be attached to anything. We can pull keys on other instruments, ring bells, even turn on lights and stuff. The keymaps in Music-X are rather like this.

About actions

A Keyboard Map is something that uses Note events to trigger other “actions”. Possible actions include the playing of MIDI events, the playing of Music-X events, and the triggering of Music-X Commands. A Keyboard Map is basically a list of the 128 possible incoming MIDI key numbers, and a corresponding list of the particular actions that should be taken when each of the keys is received. Keymaps are edited by connecting actions to key numbers.

About ranges

Each key number can be individually set to trigger its own type of action. However, in most cases, groups of keys will use the same action type. For instance, when using a keymap to “split” the mother keyboard and play notes on two different sound modules, many or all of the keys would trigger the same action—Note events. The editor allows “ranges” to be defined and edited en masse. A range is a group of adjacent keys which trigger the same type of action. A range may be as wide as the entire 128 key MIDI keyboard or as narrow as a single key. Once a range is defined, an action can be attached to the whole group at once. Ranges are defined on the miniature keyboard in the center of the Keymap Editor Page.

The face of the Keymap Editor Page

The controls of the Keymap Editor Page are divided into four basic areas. At the top is the Mode panel. The controls in this panel are used to choose the current map, change the modes that govern the way the miniature keyboard operates, and make overall editing decisions. The second area contains the miniature keyboard, used to create and edit ranges in the current map. The bottom of the screen holds the Action list and the Parameter window, used to choose the actions that will be attached to each range.

There are four basic steps to editing a keymap.

- ☐ Choose one of the four maps with the **MAP** buttons in the Mode Panel.
- ☐ Use the controls in the miniature keyboard to define a range and set its transposition.
- ☐ Use the Action list and the Parameter window to set the action to be triggered by notes in that range.
- ☐ Click **SET** to set the currently defined range and action into the map.

The rest of this chapter contains descriptions of the individual features of the Keymap Editor Page.

Title Bar

The Title Bar is found along the top of all Music-X pages to remind one where one is. In the Keymap Editor, the name of the last loaded or saved keymap file will appear here as well, after the word **File:**. If no keymap file has been saved, then the file name will be untitled, i. e., “File: Untitled.Kmap”.

The Title Bar becomes the Menu Bar when the right mouse button is pushed, showing the titles of available menus. To access the menus in the Keymap Editor, press the right mouse button and point to the desired menu title in the Menu Bar. The chosen menu will “pull down”, listing the available menu items. To choose a menu item, point to it, and release the right mouse button while the item is highlighted.

File Menu

The File menu in the Keymap Editor Page is used to save and load keymap files to and from disk. A keymap file contains the settings for one keymap. There are four keymap slots in memory. Saving a keymap file saves the settings for the current keymap. Loading a keymap file overwrites the current keymap. Keymap files are independent of map numbers. That is, keymaps created in one map slot may be saved and loaded into another map slot.

There are two approaches or philosophies when using keymaps. One is to design keymaps that are to be used with particular songs. For example, splitting the keyboard to play the bass line and a solo sound live with a sequenced background, or attaching drum fills to particular keys. In that case, keymaps should be saved with the song in a performance file, from the Sequencer Page. Saving a performance file with the **Keymaps** subfile switch turned on causes all four keymaps to be saved along with the performance.

The second approach to using keymaps is to design generalized keymaps that will be useful for many songs. These keymaps can be kept on disk as tools, and loaded as needed during editing sessions to facilitate the editing process. For example, transposing the mother keyboard to record parts beyond the upper or lower limits of the keyboard, or triggering sequences from the keyboard to arrange the form of the song in real time. In this case, it is better to save and load keymaps individually from the File menu of the Keymap Editor Page. By loading a keymap, changing the current map slot, and loading another keymap, keymaps can be “mixed and matched”. In general, save a keymap separately if you think it will be handy for building other compositions, creating songs, or editing in general.

Load...

(RightAmiga-“L” from the keyboard)

Load is used to load a previously saved keymap file from disk into the current map slot, overwriting its previous contents.

Choosing **Load** from the File menu first calls the file requester under the guise of **Load Keymap**. If you decide not to load, click **CANCEL** to exit the file requester and return to the Keymap Editor. Otherwise use the file requester to find the correct drawer and the file you want to load. Then click **OK**. The keymap file will be loaded into the current map slot.

[see File Requester chapter]

Save...

(RightAmiga-“S” from the keyboard)

Save is used to save the contents of the current keymap to disk in a keymap file.

Choosing **Save** from the File menu calls the file requester titled, in this instance, **Save Keymap**. If you decide not to save, click **CANCEL**. Otherwise use the file requester to locate the directory your file is to be saved to. Then use the **File:** gadget to name your file. The suggested file extension for keymap files is “.Kmap”. If you are resaving a previous file, use the same name to overwrite the old file with the new file. Then click **OK** to save the Keymap file.

Exit

(RightAmiga-“X” from the keyboard)

Choosing this item exits the Keymap Editor Page and returns you to the Filters Page. If edits were made to the current map, they will be made permanent by exiting.

♦ **Note:** You can also exit the Keymap Editor by clicking the **OK** button in the top panel.

Mode Panel

At the top of the Keymap Editor Page is the Mode Panel, used to change the overall editing modes. The panel is divided into three sections. In the section at the left are the three decision buttons that allow you to confirm the edits made to the current map, or change your mind and reverse them. In the middle section are the three buttons used to set the editing mode for the miniature keyboard. In the section at the right end of the panel are the **Map** buttons, used to select the current map. Descriptions of the controls in the Mode Panel follow.

UNDO

(“U” from the keyboard)

When clicked, this button reverses the last change made to the current map.

Example: Using **UNDO** to correct to recover from a destructive edit. (This example uses concepts discussed later in the chapter.)

- ☐ Create three new regions across the map, clicking **SET** after each change.
- ☐ Create a fourth range that overlaps the three previous regions, and click **SET**.
- ☐ Click **UNDO**. The last region is erased, and the three previous regions are restored.

[see Keymap Editor / Mode Panel / RANGE]

[see Keymap Editor / Mode Panel / SET]

[see Keymap Editor / Mode Panel / UNDO]

CANCEL

("C" from the keyboard)

CANCEL acts like a super undo. When clicked, this button cancels all changes made to the current map. The map is returned to the condition it was in when you started editing it, no matter how many changes have been made, as long as you stay in the current map. Changing the current map to one of the other four keymaps makes any changes permanent.

Example:

- ☐ Enter the Keymap Editor Page.
- ☐ Choose map 1.
- ☐ Edit map 1.

- ☐ When satisfied, choose map 2.
- ☐ Edit map 2.
- ☐ Click **CANCEL**.
- ☐ Map 2 is returned to its original condition; all changes are reversed. Map 1 is not affected.

◆ Note: If you use **CANCEL** accidentally, you can use **UNDO** to undo the cancel.

OK

("O" from the keyboard)

Clicking **OK** sets the changes made to the current keymap and exits the Keymap Editor Page. You are returned to the Filters Page. Clicking **OK** is the same as choosing **Exit** from the File menu.

SHOW

("H" from the keyboard)

Clicking the **SHOW** button puts the Keymap Editor into "Show Mode". Show mode is used to show the settings that already exist within the map.

When the editor is in Show mode, pressing a key on the mother keyboard causes the editor to display the action attached to that key. Optionally, a key can be chosen by clicking in the black area directly underneath the miniature keyboard.

RANGE

("R" from the keyboard)

Clicking the **RANGE** button puts the Keymap Editor into "Range mode". When in Range mode, ranges can be defined by pressing keys on the mother keyboard.

Example: To define a range from the mother keyboard, first click the **RANGE** button to enter Range mode. Then press the two keys at each end of the range, one after the other. That is, press the lowest key within the desired range and the highest key within the desired range. If you make a mistake, press the keys again. The current range is always calculated using the last two keys pressed. If the range is to be one key wide, press that key twice. The miniature keyboard will show the range highlighted in purple. After defining a range, define the action for that range by selecting an event type and parameter (using the Action list and Parameter window, discussed later), and setting the pivot value if needed. Then click the **SET** button to set the new range into the current keymap.

SET

("S" from the keyboard)

Clicking the **SET** button sets the currently defined range and action into the current keyboard map.

MAP 1 2 3 4

("1", "2", "3" and "4" from the keyboard)

At the right end of the mode panel are the **MAP** buttons. These buttons are used to choose which of the four keyboard maps will be edited by the rest of the controls in the Keymap Editor Page. Clicking on one of the **MAP** buttons will change the current map and cause the new map to be displayed in the miniature keyboard area.

Miniature Keyboard

Running across the middle of the Keymap Editor Page is the miniature keyboard display, with its collection of controls. This gadget is used to edit the ranges in the current keyboard map. The miniature keyboard is 10 1/2 octaves wide, and represents the total range of MIDI key numbers.

Following are descriptions of the features of the miniature keyboard.

Octave Scroll Bar

Just above the miniature keyboard is a scroll bar, used to bring different portions of the miniature keyboard into view. Seven octaves can be viewed at once (approximately the width of a grand piano). Clicking on the arrows at either end of the scroll bar will move the keyboard by single octaves.

Range Delimiters

The black area directly above the keys of the miniature keyboard is used to hold the range delimiters. When a range is set, using the **SET** button, two small, yellow lines appear at either edge of the range to show its limits. As more ranges are set, more delimiters appear. When a new range is set that overlaps a previous range, the previous delimiters will be erased and new delimiters will be added to show the new range.

Keys

The keys of the miniature keyboard are used to display and define a range before being set into the keymap. Ranges are defined one at a time and then set into the map before the next range is defined.

To define a range on the miniature keyboard, click and hold the left mouse button while the mouse pointer is positioned over the lowest key in the range. The key will turn purple to show that it has been selected. Then, while holding the mouse button, drag the pointer across the keys of the miniature keyboard towards the other end of the range. The keys will turn purple as you drag the pointer across them to show that

they are included in the range. When you reach the last key, release the mouse button. The range is now defined, but not “set”. After using the rest of the controls to define an action for the new range, click the **SET** button to set it into the keymap.

Alternately, when in Range mode, a range may be defined by pressing any two keys on the mother keyboard.

Pivot markers

The black area directly below the keys of the miniature keyboard is used to hold the pivot markers. The pivot markers are used to indicate and to set the transpositions of Note events and Play Sequence events when used in a keymap. (When used with other actions, the pivot markers can be ignored.) Pivot markers are displayed as small triangles with stretchable tails (the tails are sometimes invisible when not stretched). There are two pivot markers: one red, and one yellow. The yellow marker only appears in Show mode. Each pivot marker behaves differently when operating in Show mode, or Range mode, and when used with Note events, or Play Sequence events. The two markers, and their use in the two editing modes, are discussed below; Range mode is discussed first, then Show mode.

While the Keymap Editor is in Range mode, the yellow pivot marker is dormant, and is not displayed. The red pivot marker has two functions:

The red pivot marker’s first function in Range mode, is as a transposition marker when used with a range of Note events. Transposition for a range of Note events is set by placing the red pivot marker under *the key to be triggered by the bottom key in the current range*. To indicate the connection between the bottom note in a range, and the target note, the red pivot marker’s tail remains attached to the bottom note in the current range. The pivot marker can then be placed under any of the 128 keys in the miniature keyboard by clicking in the black area below the keyboard. The tail stretches from the base of the range to the target note where the pivot marker is. The length of the tail gives a general indication of the magnitude and direction of the transposition. Placing the marker exactly at the base of a range sets the transposition for that range to zero, and causes the tail to disappear.

Keymap Editor / Miniature Keyboard

Example: Using the red pivot marker to transpose the mother keyboard down one octave.

- ☐ From the Filters Page, turn on the **RE-OUT** switch button for the channel your mother keyboard is transmitting on so that we can hear note events echoed to MIDI OUT while editing in the Keymap Editor Page.
- ☐ From the Keymap Editor Page, click the **RANGE** button to put the editor into Range mode.
- ☐ Define a range in the miniature keyboard by pressing the lowest and then the highest keys on the mother keyboard. The range of the mother keyboard should now be highlighted in purple in the miniature keyboard.
- ☐ Choose **Play Note** from the action list in the bottom left corner of the page to attach that action to the currently defined range.
- ☐ In the parameters window (see below), choose the MIDI channel that the chords should be played on (the channel that your piano module is set to receive on).
- ☐ Now click in the pivot area below the miniature keyboard to place the triangle one octave below the bottom key. If the lower octave is not in view in the miniature keyboard window, use the scroll bar above the miniature keyboard to bring it into view before clicking.
- ☐ The tail of the red pivot marker should be attached to the bottom note in the range, and the head of the marker should be stretching down one octave.
- ☐ Click **SET** at the top of the page to set the new range into the map.
- ☐ The map is now ready to use, for instance, to record a bass part one octave below the range of the mother keyboard.

The red pivot marker's second function in Range mode, is to set the transposition for a range of Play Sequence events. Unlike Note events, Play Sequence events have an original key at which the sequence is played normally. Each key within a range of Play Sequence events plays the same sequence at a different transposition. It's more useful to describe the transposition of the range as it relates to the original key. This is done using a "pivot value". The pivot value represents *the key that would play the sequence with no transposition*. The pivot value for a range of Play Sequence events is set by placing the red pivot marker underneath one of the keys in the miniature keyboard. Placing the red pivot marker in the center of a range of Play Sequence events creates a range where the center key plays the sequence in its original key, higher keys play the sequence in higher transpositions, and lower keys play the sequence at lower transpositions. This is why pivot markers are called "pivot markers"—the range "pivots" around the center. You can, however, place the pivot point anywhere, including outside of the range.

Example: Setting up a keymap to play an automated "one finger" bass line in various keys.

- ☐ In the Sequencer Page, record and store a simple, one or two measure bass pattern. This will be the source sequence for our Play Sequence events.
- ☐ Turn the source sequence off, so that it can't play by itself.
- ☐ In the Filters Page, turn on one of the keymaps for the channel that the mother keyboard is transmitting on (let's say map 1).
- ☐ In the Keymap Editor, choose map 1 as the current map.
- ☐ Click **RANGE** to put the Keymap Editor into Range mode.
- ☐ Drag the mouse across the miniature keyboard to define a range covering the bottom two octaves of the mother keyboard.

- ☐ Click **Play Sequence** in the Action list to attach Play Sequence events to the currently defined range.
- ☐ In the extra buttons that appear when **Play Sequence** is chosen, choose **Cut Sequence** (this is described in detail later).
- ☐ In the Parameter window, in the bottom right corner of the Keymap Editor Page, choose the source sequence with the bass pattern in it.
- ☐ Click in the pivot area below the miniature keyboard to place the red pivot marker in the middle of the range. This defines the key that will play the sequence with no transposition.
- ☐ click **SET** to set the current range in the keymap.

When the Keymap Editor is in Show mode, the red and yellow pivot markers are both displayed. The Keymap Editor is put into Show mode by clicking **SHOW** in the mode panel. Show mode is used to show the actions that are already attached to the map. This is done using the concept of the “current key”. The current key can be chosen by pressing a key on the mother keyboard or by clicking underneath the keyboard. Once the current key is chosen, the Action list and the Parameter window will configure themselves to show the action attached to that key. By playing a chromatic scale on the mother keyboard and watching the action list, one can see what’s in the map.

The red marker has three functions in Show mode:

First, it can be used as it is in Range mode to set the transposition for the current range (drag the mouse across the keyboard to define a range, then click underneath the keyboard to define the key that will be played by the bottom key in the range).

Second, it can be used as it is in Range mode to set the pivot value for a range of Play Sequence events (drag the mouse across the keyboard to define a range, then

click underneath the keyboard to define the key that will play that sequence with no transposition).

Third, it is used to show the current key. Each time a key is played on the mother keyboard, the red pivot marker moves underneath the corresponding key on the miniature keyboard. The rest of the controls configure themselves to show what action is attached to that key. When the red pivot marker is used in this capacity, its tail should be ignored.

The yellow pivot marker has two functions in Show mode:

First, when used with Note events, the yellow marker shows the transposition for the current key (pointed to by the red marker). The tail of the yellow pivot marker is always attached to the red pivot marker; to the current key. The yellow marker appears to stretch away from the red marker and point to the new note played by the current key after transposition. If the current Note event has no transposition, both markers will point to the same key; the yellow marker will be placed exactly on the red marker, hiding it. As keys are played on the mother keyboard, the markers move about to show the key received, and the key played after transposition.

Second, when used with Play Sequence events, the yellow marker shows the pivot value for the current key. The yellow marker points to the key that would normally play the sequence in its original key.

Octave Blocks

Just below the miniature keyboard is a row of blocks showing which octaves are currently being viewed. Each octave ranges from a C note up to the next B note. The C two octaves below middle C (the bottom key of the standard 61-key organ or synth keyboard) is considered the bottom note of octave number 1. The octaves above that are numbered 2 through 8 with middle C in the 3rd octave. The octave below octave 1 is octave 0, and below that, the prime octave and the double prime octave. The prime octave is denoted by a single quote sign, and the double prime, by

a double quote sign. As the miniature keyboard is scrolled to bring different portions into view, the octave blocks will show the new octave numbers.

Range:

Underneath the octave blocks is a panel containing three (alphanumeric) indicators. These are used to verify the current range and pivot value as they are changed in the miniature keyboard.

The first indicator is titled **Range:** and shows the note names and octaves of the lowest and highest keys in the current range.

MIDI Key#:

The second indicator is titled **MIDI Key#** (MIDI key number), and shows the key numbers for the lowest and highest keys in a range. Possible MIDI key numbers range from 0 to 127. Middle C is MIDI key number 60.

Pivot Value:

The third indicator is titled **Pivot Value** and shows the note name for the current pivot value. In the case of Note events, the pivot value shows the name of the note that will sound when the bottom key in a range is played, after transposition. In the case of Play Sequence events, the pivot value shows the name of the note that will play the sequence at no transposition.

Action List

In the bottom left corner of the Keymap Editor Page is the “Action list”. This list shows the possible actions that can be initiated when a key is received. Each action may have additional parameters that are listed in the Parameter window (see below). Possible actions include the sending of certain MIDI events (Notes, Control Changes, and Program Changes), the sending of certain Music-X events (Play Sequence, Mute Track, Solo Track, Change Tempo, and Change Keymap), and the initiation of certain Music-X commands. Music-X commands are not events, they are rather like key equivalents. Music-X commands cause the program to react as if

certain screen controls were used. Using Music-X commands in a keymap allows you to control Music-X from the mother keyboard as if you were using the mouse or the Amiga keyboard. The possible Music-X commands are Start Sequencer, Stop Sequencer, Time Step, and Punch In/Out. Music-X commands are used immediately and cannot be recorded as MIDI events and Music-X events can be.

The button next to each action is used to choose that action. When an action is chosen, its button will appear highlighted in yellow, and the Parameter window, in the lower right corner of the page, will change to show the range of values for the parameter associated with that action. All other buttons will be turned off.

[see Keymap Editor / Parameter Window]

Following are short descriptions of the action types. In all cases except the Music-X commands, more detailed descriptions of the various event types can be found in the Bar Editor and Event Editor chapters.

[see Bar Editor / Event Types]

[see Event Editor / Event Types]

Play Note

This action will cause a Note event to be played when a Note event is received. Note events may be transposed using the pivot markers. The Parameter window is used to choose the new MIDI channel.

◆ Note: Since any input channel can be filtered through a keymap from the Filters Page, notes on any channel can be used to trigger note actions from the keymap. However, the triggered notes will always occur on the MIDI channel specified in the keymap.

◆ Note: When trying out note actions in the Keymap Editor Page by playing the mother keyboard, one of the **Data Echo**: switches must be turned on in the Filters Page for the channel that the mother keyboard is sending on. Data being passed through the keymap returns to the Filters Page and through the Data Echo switches before being echoed.

Play Sequence

This action will cause a Play Sequence event to be played when a Note event is received. Play Sequence events may be transposed using the pivot markers. The Parameter window is used to choose the sequence number.

When Play Sequence events are chosen as the current action, three extra buttons appear between the Action list and the Parameter window. These are used to choose the cut priority parameter for the Play Sequence event. The choices are:

No Cut
Cut Sequence
Cut Notes

No Cut lets the sequence finish naturally when the key is released. **Cut Sequence** stops the sequence but lets any currently playing notes finish naturally. **Cut Notes** stops the sequence and cuts off any currently playing notes as soon as the key is released.

Control Change

This action triggers a Control Change event. The Parameter window offers a list of the 128 possible controller numbers. The value for the Control Change event is actually converted from the attack velocity of the key that triggers it. This means you can do things like send MIDI volume (controller 7) commands by hitting the mother keyboard harder or softer.

When Control Change events are chosen as the current action, a row of numbers from 1 to 16, titled **Channel:**, appear above the Parameter window. These are used to choose the MIDI channel for the triggered Control Change events.

Example: Using a keyboard map to record Control Change events to send MIDI volume information to a non-velocity sensitive keyboard or sound module. (The module must support MIDI volume—controller 7—for this example to work; the CASIO 101 is one that does.)

- ☐ Record a part for the sound module with some rests in it (a “horn stabs” part will do well).
- ☐ Use the miniature keyboard to define a range across the mother keyboard (size is not important).
- ☐ Choose **Control Change** in the Action list.
- ☐ Choose **Control 007** in the Parameter window (this is MIDI volume).
- ☐ Click in the row of numbers titled **Channel** to choose the channel that the part was originally recorded on, and that the sound module will be receiving on.
- ☐ Click **SET** to set the range into the current keymap.
- ☐ Click **OK** or choose **Exit** from the File menu to exit the Keymap Editor.
- ☐ From the Filters Page, click the channel button that the mother keyboard will be transmitting on, to choose the current filter for the mother keyboard.
- ☐ Turn on the **Keyboard Map** button corresponding to the keymap that our special map is contained in (1 through 4). This assures that MIDI Note On and Note Off messages from the mother keyboard will be sent through the keymap before being recorded.
- ☐ Turn on the **Data Echo** switch labeled **RE-OUT** so that what we are about to record will be sent to the sound module as we play.
- ☐ From the Sequencer Page, record a new sequence to contain the controller information.

- ☐ As the pre-recorded “horn stabs” part plays back, tap along on the mother keyboard. The note events from the mother keyboard will be transmuted into MIDI volume messages. You’ll find that the greater velocity you apply to the keys, the louder the horn part will get. By anticipating each stab with a tap of the right velocity, loud and soft effects can be imparted on the part.
- ☐ When the song finishes, store the new part to a sequence. Then Merge the new sequence with the original horn part. *Voila!* Velocity type effects on a non-velocity sensitive keyboard.

Program Change

This action triggers a Program Change event. The Parameter window is used to choose the program number. Possible program numbers range from 1 to 128.

When Program Change events are chosen as the current action, the row of **Channel** numbers appears above the Parameter window. Click on them to choose the MIDI channel for the triggered Program Change events.

Mute Track

This action triggers a Mute Track event. The Parameter window is used to choose the track number to be muted. Track numbers range from 1 to 20.

Solo Track

This action triggers a Solo Track event. The Parameter window is used to choose the track number to be soloed. Track numbers range from 1 to 20.

Example: Recording a “dub remix” version of a prerecorded performance.

- ☐ From the Sequencer Page, load a performance to be remixed.
- ☐ From the Keymap Editor, load “Mute&Solo.Kmap” from the “Kmaps” directory of the Music-X Examples Disk into one of the map slots.

- ☐ In the Filters Page, set the filter for the channel that the mother keyboard is transmitting on to use the keymap that you loaded “Mute&Solo.Kmap” into.
- ☐ Record a sequence and play around on the mother keyboard. You’ll discover that the left hand will mute tracks and the right hand will solo them. The resulting sequence containing only Mute Track and Solo Track events can be stored and used as an “auto mix” control sequence.

Music-X Command

This action triggers Music-X commands in response to incoming MIDI Note messages. Four different Music-X commands can be used. The Parameter window lists the possible commands. These are described below.

Start Sequencer

This command starts the sequencer as if the **PLAY** button in the Sequencer Page or Editor Pages was clicked.

Stop Sequencer

This command is the equivalent of clicking the **STOP** button in the Sequencer Page or Editor Pages.

Time Step

This command can be used when step recording in the Bar Editor or Event Editor to step the clock forward as if the **STEP** button was clicked. Using this command lets you Step record while keeping your hands on the keys.

Punch In/Out

This is like pressing the **RETURN** key while recording in one of the Punch-In modes in the Sequencer Page. This command can be used to turn one of the keys on the mother keyboard into a “remote record switch”.

Change Keymap

This command will trigger a Change Keymap command. The Parameter window lists the possible keymaps.

[see Sequencer Page / Transport Window]

[see Bar Editor / STEP and BACK]

Parameter Window

At the bottom right of the Keymap Editor Page is the Parameter window. The Parameter window is used to display extra values for the action list. That is, as different actions are chosen in the Action list, related parameters for those actions will appear in the Parameter window. For instance, when the current action is “Play Note”, the Parameter window offers the list of MIDI channel numbers that notes can occur on. Choosing one of the values in the parameter list sets a value for that parameter and that action into the current range. (To set the range into the map permanently, click the **SET** button.)

The values in the Parameter list are chosen by clicking on them. The scroll bar at the left of the window can be used to bring hidden portions of the list into view.

MIDI Recording

Illustration 10.2 shows the possible data flow paths during recording, using Filters and Keymaps. Data from the mother keyboard enters through MIDI IN. It is split into two streams. Note events go one way, and everything else goes to the Filters. Depending on the status of the Keymap switch, Notes are then sent either to the Keymaps or back to the Filters with everything else. If Notes are sent to the Keymaps, they are transmuted into new events. MIDI events thus produced are sent through the Data Echo switch, while Play Sequence events trigger the playback of new sequences. Events pass through the Filters and are captured in the Record Buffer. If the RE-OUT switch is on, they are also echoed to MIDI OUT. Sequences

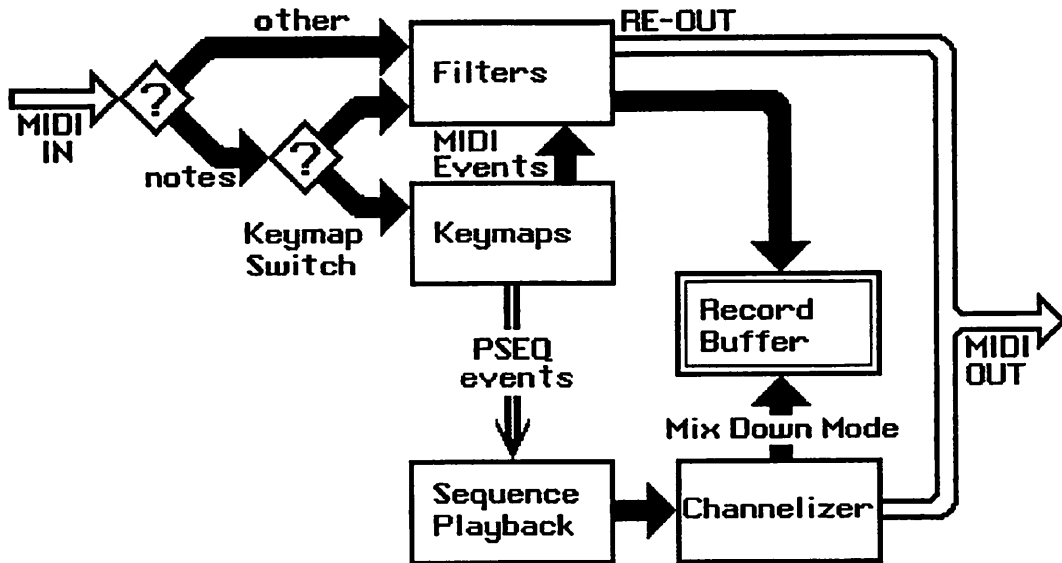


Illustration 10.2 MIDI recording data flow paths

playing during record are passed through the output channelizer and echoed to MIDI OUT. If Mix Down Mode is on, data from sequences playing back is also captured in the record buffer. In the end, the record buffer can contain quite different data than was played on the mother keyboard!

Try loading and experimenting with the following example keymaps which can be found on the Music-X Examples disk, in the Kmaps directory:

Overlap.Kmap

This keymap transposes the lower half of the keyboard up into the range of the upper half and sends them both out on channel 1. Interesting effects can be achieved by playing both hands in the same range. For instance, super fast repeating chords, or guitar-like strumming. It can also be useful for playing chromatically moving horn voicings.

Backwards.Kmap

This one turns the mother keyboard upside down. The top MIDI note (note 127) plays the bottom MIDI note (note 000), note 126 plays note 001, etc. The middle of the mirror occurs between E Flat and E natural above middle C (MIDI notes 63 and 64). The notes are played on MIDI channel 1. Try loading this one and using it to overdub an upside down, right-handed bass line against a prerecorded sequence. It's guaranteed to strip away preconceptions and give you some new ideas!

Backwards2.Kmap

You might have noticed that the standard piano keyboard layout is physically symmetrical around D natural or A flat. This keymap exploits that fact; it's mirrored around middle D (D above middle C, MIDI key 62). That gives it the property of making the keyboard feel almost "normal". That is, all the white notes are still in the key of C, and all the black notes still play an F# major pentatonic scale. All the D's are D's and all the A flat's are A flat's. The only thing different is that it's upside down! Using this keymap, you can play the pieces you know in the correct keys, using the fingering patterns you know, as long as you reverse the roles of your hands. Here's your chance to prove your ambidexterity!

Sequences1-61.Kmap

This keymap is set up to play the first 61 sequences in memory from a standard 61-key MIDI keyboard. Each key triggers a different sequence. The sequences are played in their original keys. Load this one up during an editing session when you need to record Play Sequence events.

Chapter 11 - Amiga Samples Page

Overview.	319
Title Bar.	321
Mode Menu.	321
File Menu	321
Options Menu	324
Sample List.	325
Edit Panel	327
Creating Envelopes	332

Amiga Samples Page

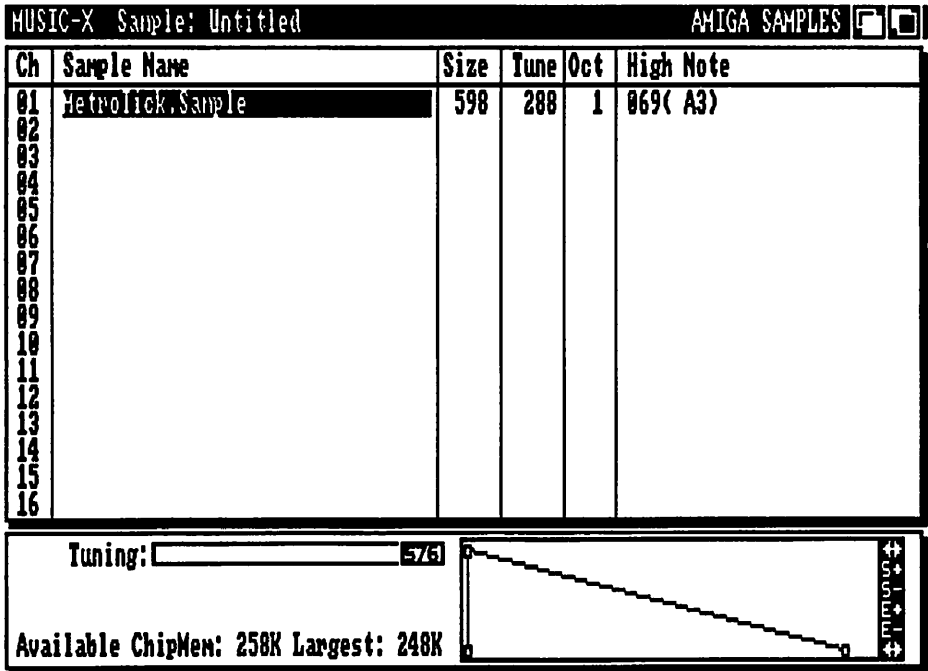


Illustration 11.1 Amiga Samples Page

Overview

The Amiga Samples Page is accessible from the Mode menu available in each of the four main pages, (Sequencer Page, Filters Page, Amiga Samples Page, and Librarian Page), by choosing **Amiga Samples** from the menu or by using the key combination, RightAmiga-“A”.

The Amiga Samples Page is used to manage internal Amiga Samples. Samples in IFF and Sonix formats can be loaded from disk into memory to be played by the sequencer. Each sample can be shaped by its own 16 stage envelope and retuned. Samples may be resaved, with their envelopes and tunings, in IFF format.

What is a Sample?

For people who are unfamiliar with the term, a “sample” is a digitized sound. Sampled sounds can sound like anything that can be recorded.

To digitize something merely means to store it in digital format. Sounds, as we hear them, are very fast fluctuations in air pressure. Using microphones and electronics, it's possible to convert fluctuations in air pressure to fluctuations in voltage. Then, using something called an “analog-to-digital converter”, the relative intensity of the fluctuation can be measured at any point in time and turned into a number. By measuring the fluctuations and noting their intensities many times per second, we can store a series of numbers that represent the history of the fluctuations over time. Then later, using something called a “digital-to-analog converter”, the series of numbers can be converted back to voltage fluctuations, and again, back to sound waves that we hear. The term “sampling” comes from the process of measuring the sound in a series of short, quick, samples.

For the techies, the Amiga has four built-in 8-bit digital-to-analog converters that can each read the same or different areas of memory simultaneously using “Direct Memory Access”, independently of each other and of the processor. This means the Amiga can play four samples at once.

Music-X allows you to store up to 16 individual samples in memory and play them via the the mother keyboard, or the sequencer. While there can be 16 samples in memory, no more than four can be sounding simultaneously. If a fifth sample is triggered while four are playing, one must be dropped.

The rest of this chapter will describe the features of the Amiga Samples Page.

Title Bar

The title bar in the Amiga Samples Page shows the name of the page and the name of the last loaded or saved sample file. If no file activity has taken place, the filename will be untitled. For example: "Sample: Untitled".

The Title Bar becomes the Menu Bar when the right mouse button is pushed, showing the titles of available menus. To access the menus in the Amiga Samples Page, press the right mouse button and point to the desired menu title in the Menu Bar. The chosen menu will pull down, listing the available menu items. To choose a menu item, point to it, and release the right mouse button while the item is highlighted.

Mode Menu

The Mode menu available from the Amiga Samples Page is the same menu available from the other main Music-X Pages. It is discussed in detail in the Sequencer Page chapter.

[see Sequencer Page / Mode Menu]

File Menu

The file menu is used to load and save sample files from and to disk. Sample files are loaded and saved individually using the current sample entry in the sample list (see below). Multiple sample files can be saved and loaded in performance files from the file menu in the Sequencer Page.

[see Sequencer Page / File Menu / Load Performance]

[see Sequencer Page / File Menu / Save Performance]

Load Sample

(RightAmiga-“I” from the keyboard = Load IFF Sample)

(RightAmiga-“X” from the keyboard = Load Sonix)

The **Load Sample** item has a pop out menu containing two items. These items are:

IFF...

SONIX...

These two items are used to load a sample from disk into the current sample list position. The first item loads any sample in “IFF 8SVX” format. IFF stands for “Interchange File Format” which is set of common file standards for the Amiga and other computers. “8SVX” is the format for 8-bit sampled voices. Sonix samples are stored in another format and usually have a file extension of “.ss”. Sonix samples loaded into Music-X may be saved again in IFF Format (see Save below). Each item in the pop out menu calls the file requester, appropriately titled, respectively:

Load IFF Sample

Load Sonix

The file requester can be used to locate the desired sample file and then load it into the current sample list position. If you decide not to load the file, click **CANCEL** in the file requester, otherwise, click **OK**.

Save IFF Sample

(RightAmiga-“V” from the keyboard)

This item is used to save the current sample, with its envelope and tuning settings, to disk in a sample file in IFF 8SVX format.

Choosing this menu item calls the file requester with the title, “Save Sample”. If you decide not to save, click **CANCEL**. Otherwise, use the file requester to locate

the disk and drawer to save your sample file to. Then name your file. The recommended file extension for sample files is “.Sample”. Then click **OK** to effect the save. Music-X also saves an icon for the sample file in the same directory. The icon for a sample file looks like a miniature CD.

Free Sample

Choosing this menu item erases the current sample from the sample list. The **Available ChipMem** indicator will increase as the memory is freed.

[see Amiga Samples Page / Edit Panel / Available ChipMem:]

Free Largest Octave

When a sample is recorded, it is recorded at one pitch. When a sample is played back it is played at many pitches. This is done by having the hardware read through the sample at various rates (similar to changing the speed of a tape machine during playback). This is called “stretching” a sample; playing it at other than its original pitch. Samples can only be stretched so far before they begin to sound really funny. The solution is to use more than one sample, sampled at different original pitches. That way notes can be played by samples whose original pitches they are closest to without stretching them too much. This process is called “multi-sampling”.

IFF and Sonix samples both follow the convention that multi-samples are to be spaced at one octave intervals, and each lower octave sample will use twice the memory of the next highest octave. Usually, because of the difficulty of sampling with strict memory constraints, the lower octaves of sampled sounds have been synthesized rather than resampled, and are exact reproductions of the higher octaves, spread over twice the memory so that the hardware can play them at a high enough rate to have decent fidelity.

For all practical purposes, when dealing with IFF and Sonix samples, we can call multi-samples “octaves”. When a sample is loaded into Music-X, the sample list **Octave** column is used to show the number of octaves in the sample. Most sampled sounds will contain more than one octave. If you don’t plan to use the lowest

octaves in your composition, you can save memory by erasing just those octaves and freeing the memory they were occupying. That is what **Free Largest Octave** is for.

Choosing **Free Largest Octave** from the file menu causes the lowest (and therefore largest) octave of the sample to be erased, and the same amount of memory to be freed. The **Octave** column will decrease by one. Repeatedly choosing this menu item strips more octaves off the bottom. The minimum number of octaves a sample can have is one. Freeing the largest octave when there is only one octave left will free the sample.

♦ Note: If you do free a few octaves of a sample, and want to save the modified sample, consider renaming it, and keeping the original in case you find out you need those octaves later.

Options Menu

The Options menu in the Amiga Samples Page has one item.

Audio Filter

Choosing this menu item turns the Amiga's built-in audio filter on and off. The Amiga hardware will verify the status of the filter by brightening the power LED ("Light Emitting Diode"—the power light) when the filter is on, and dimming it when it is off. The filter rolls off the high frequencies starting at around 4k in a gradual slope. No frequencies should be heard above 7k. This helps get rid of possible aliasing noise when playing internal samples. In most cases the filter is not needed and greater fidelity can be achieved by turning it off (leaving the menu item unchecked).

♦ Note: In the Amiga 1000, the filter is not software controlled and therefore stays on regardless of menu-item status or LED intensity.

Sample List

(CursorUp moves up through sample list)

(CursorDown moves down through sample list)

The most prominent feature of the Amiga Samples Page is the sample list. The list has 16 entries for 16 samples, divided into columns of information that say things about the samples. Samples are loaded into the list, and altered, using the tuning slider and envelope editor in the edit panel at the bottom of the page. As in other editors in Music-X, editing is done to the “current” item. In the Samples Page, this is the “current sample”. The current sample is identified by a bright blue highlight bar in the **Sample Name** column of the sample list. The current sample can be chosen in two ways. First by clicking on any entry in the sample list, and second by using the CursorUp and CursorDown keys to move the highlight bar through the list.

Along the top of the sample list, written in light blue, are the headings that indicate the type of information displayed in each column. (If a list entry is empty, it will appear blank in all columns.) Following are descriptions of the columns, listed by heading, as they occur on the screen, left to right.

Ch

Read this heading as “Channel number”. It denotes both the position of the sample in the list, and the channel number for that sample.

When samples are loaded into memory, they are placed in a list and identified by their position in that list. When Music-X uses samples to play a sequence, it treats the list positions as channel numbers. That is, if a sequence contains Note events on MIDI channel 1, and the **Out** (output bank) field of that sequence has been set to **Int** (internal voices), then those note events will be played using the first sample in the sample list. Notes on channel 2 would be played using the second sample in the list, and so forth.

[see Sequencer Page / Sequence List Window / Out]

What's really going on: A sample is a series of bytes. Different samples are loaded onto different sections of the Amiga's memory. There are four sound chips in the Amiga. Each can read a section of memory and convert a series of bytes (digital information) to an electrical signal (analog information). The electrical signal can then be fed into an amplifier and heard on speakers. When the sequencer (playing a sequence whose output is set to Int) finds a Note event to be played internally, it uses the channel number (1-16) attached to that event to decide which sample should be played (1-16). The next available sound chip is then told to play the section of memory containing that sample. If all the sound chips are busy, then one of them is told to stop what it's doing and start playing the new sample.

Sample Name

This field shows the name for each sample. The sample name is a display only, and cannot be changed directly. Samples are named in the file requester when they are saved.

Size

This field shows the amount of RAM, in bytes, used to store the sample in the Amiga. The size of the sample as shown here may differ from the size of the sample file on disk. This is because sample files on disk contain extra information (called a "header") which is stripped when the sample is loaded into memory.

The size of a sample in memory will shrink as octaves are freed, using the **Free Largest Octave** function in the file menu (see above), and will be reflected in this field.

Tune

This column shows the tuning offset used when playing the sample. The tuning offset for the current sample can be altered using the tuning slider in the edit panel (see below).

Oct

Read this column heading as “Octaves”.

This column is used to show the number of octaves in the sample. This number will decrease by one when the **Free Largest Octave** function is used (see above).

High Note

This field shows the highest MIDI key number that the sample can be played at. The upper pitch limit of a sample is determined by the Amiga’s sound chip hardware and by the number of octaves in the sample. This number will change as a sample’s tuning slider is moved. This is because as a sample is tuned down, higher key numbers will be needed to play the same pitch.

There is no lower limit to the range of keys that will play a sample, other than the range of MIDI key numbers.

Edit Panel

At the bottom of the Amiga Samples Page is the Edit panel, containing the tuning slider, and the envelope editor. These controls are used to edit the current sample. Also present are the memory indicators showing system memory available for samples.

◆ **Note:** It’s possible to play Amiga samples from the mother keyboard while editing them if a filter has been set up to do so in the Filters Page. It’s also possible

to edit Amiga samples while the sequencer is playing, to hear the edits in context, by starting the sequencer from the Sequencer Page, and returning to the Amiga Samples Page.

Descriptions of the controls in the Edit panel follow.

Tuning:

("A" from the keyboard = play sample at A440)

("C" from the keyboard = play sample at middle C)

The tuning slider is used to adjust the tuning offset for the current sample. This can be used to fine tune the pitch of the sample to match other instruments, or to transpose the sample by some amount. The slider can be moved by dragging it or by clicking on either side of the drag bar and holding the mouse button.

Each sample has a natural pitch—the one it was originally recorded at. Different pitches are obtained by playing the sample back at different rates. These rates are calculated automatically by the system to coincide with the expected pitches of Western harmony. However, if a sample was originally recorded slightly out of tune, then all the calculated pitches will also be out of tune. The tuning slider can be used to compensate for that. This number represents a number of fine tuning increments above or below the pitch at which the sample was recorded. One tuning increment represents 1/24 halfstep (4.17 "cents"). Every 24 tuning increments will change the pitch of the sample by one half step. 288 increments equal one octave. The slider ranges from -576 (two octaves below normal) to +576 (two octaves above normal).

Two keyboard equivalents have been implemented to assist the tuning process. Pressing the Amiga's "A" key plays the sample at A440 (MIDI key 69), and the "C" key plays middle C (MIDI key 60). These can be used to compare the sample's pitch against a tuning reference.

Available ChipMem:

Read this as “Available Chip Memory”.

This indicator shows the amount of room left in the Amiga’s memory for storing samples. Samples reside in an area of memory called “Chip Memory”. This is something that the Amiga operating system takes care of. This number will decrease as new samples are loaded, and increase as samples are freed. Other programs beside Music-X currently in memory may also use Chip Memory. If there is not enough Chip Memory, samples may not be loaded.

◆ **Note:** Chip Memory is separate from the Music-X “buffer” area that is reserved for sequences and editing when Music-X boots up. When a performance is loaded that includes sequences and samples, the sequences are put in the buffer area, and the samples are put into available Chip Memory. Loading samples does not decrease the number of events that can be stored in sequence memory.

Largest:

This indicator shows the size of the largest single contiguous block of available Chip Memory. Since samples must be loaded into an unbroken section of memory, this gives an indication as to the size of the largest sample that may still be loaded. This number will be smaller than the number in the **Available ChipMem:** indicator if free memory is not contiguous. This number will change as samples are loaded or freed.

Envelope Editor

The envelope editor is used to shape the volume of the sample over time, as it plays. This shape is called an “envelope”. By changing a sample’s envelope, its overall character can be changed.

Envelopes are easy to understand. What you see is what you hear. Time moves from left to right, at an even pace, and volume is represented by height. The higher parts of the envelope are the louder parts. When a sample is played, its volume will change to follow the outline of the envelope through time.

As an illustration, imagine that you have a piece of paper with a vertical slit cut out of it. Now press it up against the computer monitor just over the left edge of the envelope editor and slowly drag the paper to the right in a constant even motion. Instead of seeing an angular shape on the screen, all you see is the little bit that's visible through the slit. As you move the paper to the right, a dot appears to be moving up and down in the slit. Now imagine someone looking over your shoulder and twisting a volume knob on an amplifier to follow the movements of the dot. As the dot moves up, they increase the volume. As the dot moves down, they decrease the volume. That is the gist of what happens in the Amiga when a sample is played, and the envelope editor allows you to draw the picture that the imaginary person twisting the knob follows.

An envelope is made up of two basic components: "Nodes", and the lines that connect them. Nodes appear as small boxes with lines stretched between them. Each paired node and the line leading up to it are considered a "stage". Music-X's envelopes have 16 stages. (The first node remains fixed at zero and is not counted.) An envelope is edited by combining stages with different slopes. Slopes are adjusted by dragging the nodes. The number of stages currently being used in the envelope can be adjusted with the **E+** and **E-** controls.

One special node acts as the "sustain point". The sustain point is displayed as a darker node. The way the sustain point works is this: When you first press a key on the keyboard to play a sample, time starts moving through the envelope. As long as you hold the key down, the volume of the sample changes to match the envelope until it hits the sustain point. At that point, it goes no further. What ever level the sustain point is at—that's how loud the sound will stay as long as you hold the key. When you release the key, the envelope will continue on through the stages beyond the sustain point until it hits the last node, at which time the sound will stop. The

position of the sustain point in the envelope can be changed by clicking on the **S+** and **S-** gadgets in the column of controls along the right edge of the envelope editor.

Along the right edge of the envelope editor is a small column of controls. These are used to add or remove stages in the envelope, and to zoom and scroll the window so that different portions of the envelop can be seen. Click on the controls to use them.

Scroll Arrows

The top controls in the stack are the scroll arrows. One points left and the other points right. Clicking on either one will move the envelope in that direction, so that portions beyond the edge of the window can be seen.

S+ and S-

The next two controls are used to move the sustain point through the envelope, one stage at a time. **S+** moves the sustain point from its current node, to the next node in the envelope. **S-** moves the sustain point to the previous node.

E+ and E-

Read these as “End Plus” and “End Minus”.

These two controls are used to extend or collapse the envelope by one stage. Each time **E+** is clicked, an extra stage is added to the end of the envelope. Clicking **E-** removes the last stage. An envelope can have up to 16 stages, with 16 lines and 16 nodes, not counting the first node, which is fixed.

Zoom arrows

At the bottom of the stack of controls are the zoom arrows. Clicking on the left zoom arrow, zooms out so that more time is visible in the envelope window. Clicking on the right zoom arrow zooms in so that finer adjustments can be made.

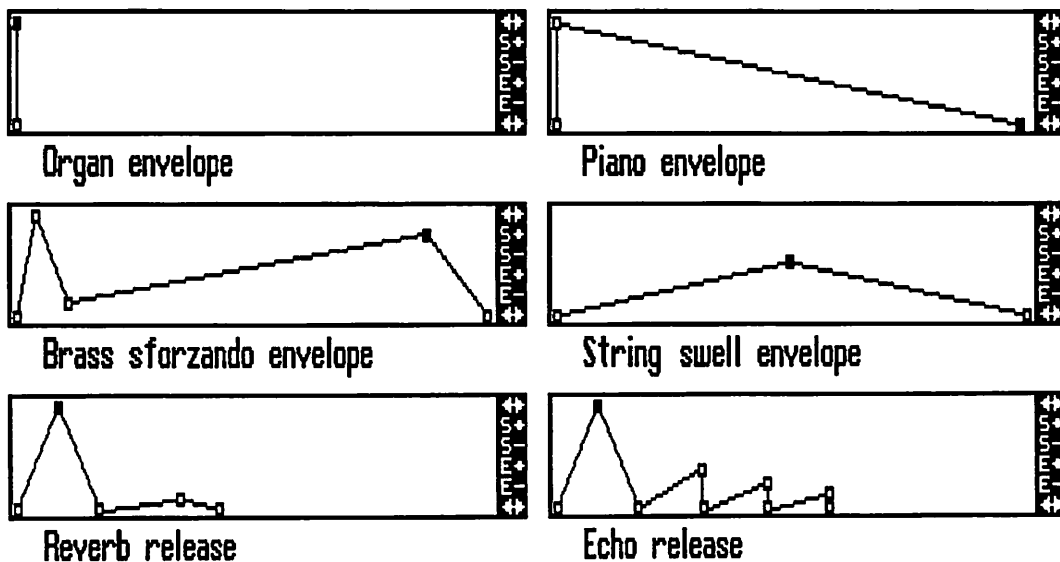


Illustration 11.2 Envelopes

Creating Envelopes

There are endless shapes that can be designed for envelopes but some general concepts can be considered while creating them. A good way to start is by recreating envelopes of existing instruments. Following are some examples.

An Organ Envelope

The first example has one stage with two nodes. The sustain point is directly above the initial node, at maximum volume. As soon as a key is pressed, the sound rises to maximum and stays there. Since the sustain point is also the end point, the sound dies away instantly as soon as the key is released.

A Piano Envelope

With this envelope, the sound rises to full volume instantly and then begins to decrease gradually towards the sustain point as long as you hold the key down. This gives the effect of a hammer initially hitting the string hard, after which the string continues to vibrate and slowly loses energy. If you release the key before the

sustain point has been reached, the sound dies away quickly, just as it would when the felt damps the string.

A Brass Sforzando Envelope

This is a more complex envelope consisting of four stages. When the key is first pressed, the sound rises quickly, but not instantly, to maximum. As soon as maximum volume is reached, volume decreases rapidly to almost zero and then begins rising slowly towards the sustain point where it stays. When the key is released, the sound dies away quickly but not instantly.

A String Swell Envelope

This envelope imitates the slower, expressive envelopes of a string section. It's good for soft background sounds. When the key is pressed, the sound begins to rise gradually towards the sustain point. Since the sustain point is at half volume, the sound stays quiet. When the key is released, the sound sneaks away gradually, not drawing attention to itself.

Reverb

With clever application, envelopes can be made to emulate effects like reverb and delay. The latter part of this envelope can be appended to any normal envelope to make it sound like reverb has been added. When the key is released, the sound dies away quickly to zero. After zero volume has been reached, and the note has effectively "ended" for the listener, volume is increased slowly to a very slight volume and then decreased slowly to nothing. This gives the impression that the sound has reflected from a distant object.

Echo

This envelope can be appended to normal envelopes to emulate a combination of reverb and delay. When the key is released, the sound dies away quickly to zero. After zero has been reached, the sound runs through a series of short ramps whose peak levels get gradually lower. This series of repeating peaks gives the effect of echo. The distance between the peaks of the ramps gives the "delay time". The rate at which each ramp rises determines how "diffused" the additional reverb effect is.

Chapter 12 - Librarian Page

Overview.	335
Title Bar.	339
Mode Menu.	339
File Menu	340
Options Menu	342
Modules Menu.	345
Mode Panel	346
Channel:.	357
Page Displays	357
Generic Patch Editor	359

Librarian Page

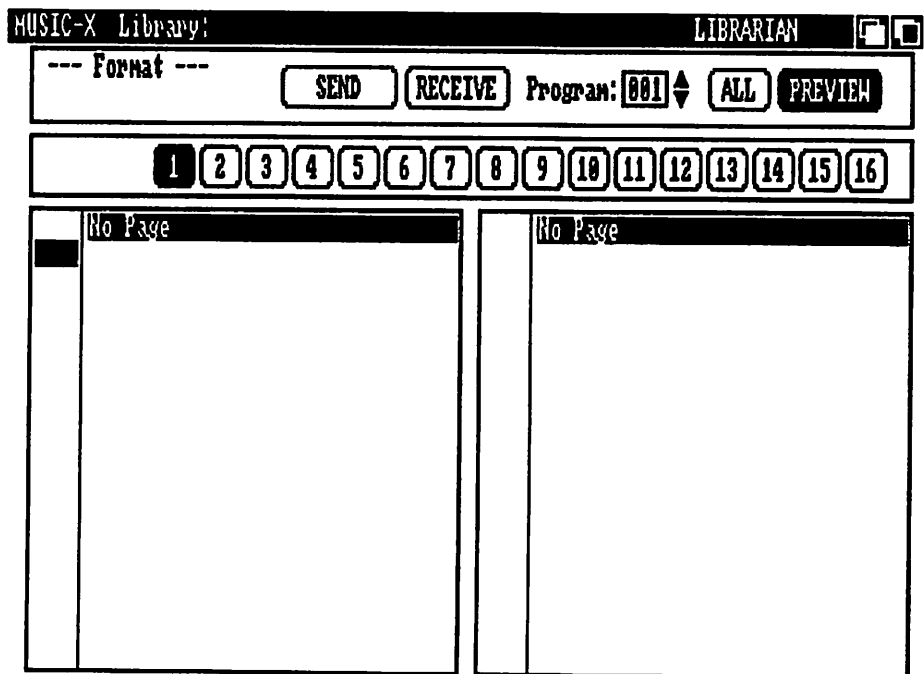


Illustration 12.1 Librarian Page

Overview

If you “boil it all down”, the Librarian Page is used to send sounds to your synthesizer.

Now, how does it do that, and how do you use it?

First, to answer those questions, a little bit of history. When synthesizers were first invented, they looked like mysterious boxes with lots of knobs, and cables stuck everywhere. By plugging in the cables and twisting the knobs, all sorts of amazing new sounds were created. When somebody came up with a sound they really liked,

they would write down the positions of all the knobs and the routing of the cables so that they could recreate that sound later. That particular combination of knob positions and cable arrangements was called a “patch”.

Well, synthesizers are still mysterious boxes, but they don’t have knobs anymore. Thanks to economics, the knobs were removed in favor of “parameters”. A parameter is rather like a knob. Where a knob has positions, a parameter has “values”. Each parameter affects one particular aspect of the sound. Sounds are created by changing the values of many parameters, one by one, until the combined effect becomes something musically usable. Then, instead of using a piece of paper to write down the positions of knobs in a patch (as in the old days), the values of the parameters that make a patch are written into the synthesizer’s memory as a “program”. A program is a string of numbers representing the values to be applied to the parameters in a synthesizer to recreate a sound. Synthesizers can typically remember up to 128 different programs. These are identified in the instrument by “program number”. Each time a program number is chosen, by hitting a button on the front panel of the synth, or by sending the synth a Program Change command via MIDI, the synthesizer resets its parameters to the values stored within that program.

So far, we’ve identified what a parameter is, what a patch is, what a program is, and what a program number is—what’s a librarian?

If you had 128 favorite programs in the world, and you programmed them all into your synthesizer, you’d be set. But what if you had 129 favorite programs, or 129,000? Where would you store the extra programs after the capacity of the synth is filled? Well, since a program is nothing but a string of numbers, and computers are great with numbers—let the computer remember the extra programs. That’s what the Librarian Page in Music-X is for. The Librarian is used to collect programs from synthesizers and send them back again when needed. This is accomplished using something called a “MIDI bulk data dump”. Most MIDI synthesizers support

bulk data dumps that allow them to send or receive portions of the data in their memory. That means programs may be transferred via MIDI between the synthesizer and the Librarian.

In Music-X, programs are collected in bundles of 16, called “libraries”. The way you use the Librarian to send sounds to a synth is to load a library, and then send one or more of the programs in that library.

An interesting complication arises. That is that each synthesizer has a different set of parameters, so they all need their own programs. Not only that, they all like you to send them programs in their own special way. Some like you to ask first, others like you to just send it, others make you say thank-you afterwards, etc. This is simplified in Music-X by something called a “protocol”. A protocol contains the rules of ceremony and etiquette for one particular synthesizer. A protocol is attached to each library, informing Music-X which synthesizer the programs in the library were designed for, and what the rules for sending those programs are. By using protocols, the librarian can collect and distribute programs for many different instruments. In fact, any device that supports some sort of MIDI bulk data dump can be used with the librarian, as long as there’s a protocol for that device. Some protocols are provided with Music-X. New protocols can also be created from scratch with the Protocol Editor.

[see Protocol Editor chapter]

In summary, the Music-X Librarian Page is used to manage programs for the synthesizers and other MIDI devices in your system. Its basic functions are to receive, collect, and send program data or other bulk data that a synthesizer or MIDI device might use.

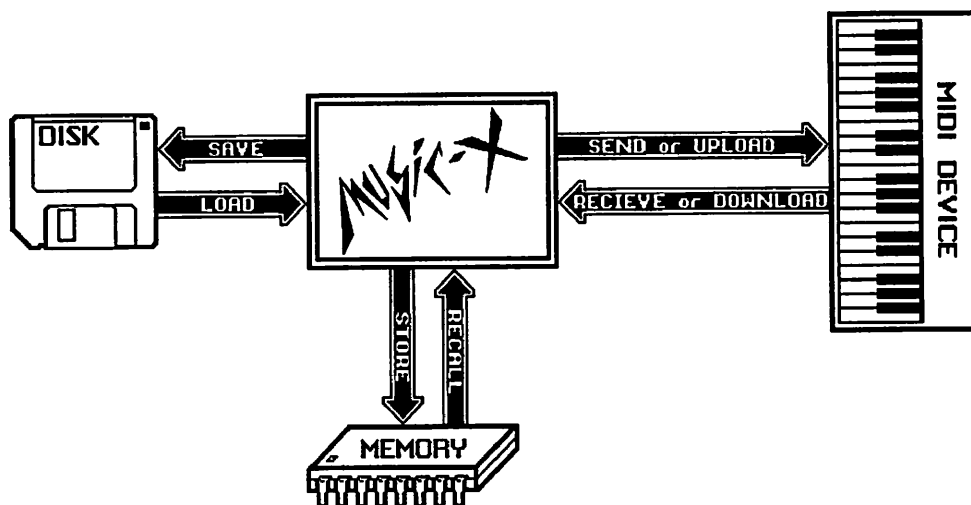


Illustration 12.2 Terminology used for moving data

The above illustration shows some general terms for moving data between Music-X and its environment. When data is copied from Music-X to disk it is being “saved”. When data is copied from disk into Music-X it is being “loaded”. When being moved from one section of memory to another, it is referred to as being “stored” and “recalled”. (Data is generally “stored” to more permanent areas after being edited in temporary areas.) The terms for moving data between Music-X and synthesizers via the Librarian Page are “sending” and “receiving”, or “uploading” and “downloading”. Sending, or uploading, is when data is moved from Music-X to a synthesizer. Receiving, or downloading, is when data is moved from a device to Music-X.

The Face of the Librarian Page

The Librarian Page is made up of four main panels. The topmost panel in the screen is called the “Mode panel”. It contains the main controls governing the movement of data in the Librarian Page, including the **SEND** and **RECEIVE** buttons. The second panel is the “Channel panel”, which is used to set the MIDI channel that communications will be transmitted over. The most prominent features in the

Librarian Page are the two panels in the the lower half of the screen, called “pages”, used to display the contents of library files. Each page holds 16 “entries” representing one data dump each (in most cases, a data dump will be a synthesizer program).

◆ Note: The controls in the Mode panel are not fixed in their meaning. Instead, they are rather like “advice” or “flags” to the protocol interpreter in Music-X. This means they can actually change function depending on the protocol being used. Since protocols are an open-ended feature, there’s no way to tell what protocols may be written in the future, or how they may affect the controls in the Librarian Page. In most cases however, the controls will function the way they are described in this chapter.

Title Bar

The title bar is used to show the name of the current Music-X Page being used. In the Librarian Page, the right end of the title bar reads **LIBRARIAN**. The title bar also gives the name of the last file loaded or saved from that page. In the Librarian Page, up to 10 library files can be resident in memory at once. The title bar will change to show the current library file being shown in one of the two page displays in the lower half of the Librarian Page.

The Title Bar becomes the Menu Bar when the right mouse button is pushed, and is used to access the menus in the Librarian Page.

Mode Menu

This Menu is available from each of the four main Music-X Pages, and is used to move between them. The features of this menu are described in the Sequencer Page chapter.

[see Sequencer Page / Mode Menu]

File Menu

The file menu of the Librarian Page is used to move individual library files between disk memory and Music-X. Up to 10 library files may be loaded into memory at once as “pages”. Individual pages may be closed to free memory when not needed.

◆ **Note:** Multiple library files may be saved or loaded in a performance file from the Sequencer Page. When a new performance is loaded, all pre-existing library files are erased from memory.

The File menu has four items. These are as described in the following paragraphs.

Load...

(RightAmiga-“O” from the keyboard)

Choosing **Load** from the file menu calls the file requester, appropriately titled **Load Library**. Use the file requester to locate your library file and click **OK**. Some example library files are included on the Music-X Examples Disk, in the “Libraries” directory. Library files end in “.Libr”.

After a library file is loaded, it will appear in the current page in the lower half of the screen.

[see File Requester chapter]

New...

(RightAmiga-“N” from the keyboard)

New is used to create a new empty library, to prepare for receiving new program data from an instrument. In order to create a new library, the type of data that the library will hold must be specified. This is done by choosing a protocol. A protocol

tells the librarian what type of data to expect, and how to communicate with the instrument that will be sending it.

Choosing this menu item calls the file requester, under the guise of **Select Protocol**. Use the requester to locate and load the protocol for the instrument or device you will be receiving from. The standard protocols released with Music-X are found in the “Protocols” directory on the Music-X program disk. Protocols are named after the instrument and data format that they apply to. For example, “D-50.Voices”.

[see File Requester chapter]

After the protocol is loaded, it is attached to a new empty library file. The new library will be displayed in the current page in the lower half of the screen. All 16 entries in the page, will be titled “empty”. The filename will be “Untitled.Lib”. The new file is now ready. At this point data may be received from the device.

Save...

(RightAmiga-“V” from the keyboard, think of second consonant)

Save is used to write the current page to disk as a library file. The protocol for that library is also saved along with the library file. (This means you can give a friend a library without having to give him or her the protocol as a separate file.) After **Save** is chosen, the file requester appears, titled **Save Library**. Use it to locate the directory to hold your file. Then name your file using the File Requester’s **File** gadget. Library names may be anything descriptive. The recommended file extension for library files is “.Lib”. Example: “KorgBasses1.Lib”. If you are resaving a previous file, use the same name to overwrite the old file with the new file. Then click **OK** to save the file..

Music-X also saves an icon for the library file, in the same directory. The icons for library files look like a miniature schematic or flow diagram, representing a synthesizer patch.

Close

(RightAmiga-“C” from the keyboard)

Close is used to close the current library. This erases it from the Amiga’s memory (library files on disk are unaffected).

To close a library, first use **Set Display** in the Edit menu to display the library in the current page (if it is not already visible). Then select **Close** from the file menu. The current library will close and the memory will be freed. The current page will then show the next available file in memory. If there are no more libraries, the page will appear empty.

[see Librarian Page / Edit Menu / Set Display]

Edit Menu

The Edit menu gives access to the Patch Editor Pages, used to manipulate the sounds in a synthesizer, and to edit the entries in a library page. Access is also given to the Protocol Editor Page where new protocols may be created. Descriptions of the items in the Edit menu follow.

Set Display

Up to ten libraries may be loaded into memory at one time. Any two of them may be displayed in the two pages in the lower half of the screen. This menu item is used to choose which of the libraries currently in memory will be displayed in the two pages. Next to this menu item is a pop-out menu that lists all the libraries currently in memory. The pop-out menu grows with the number of libraries loaded. If there are no libraries currently in memory, the pop-out menu won’t pop out! Choosing one of the libraries in the list causes that library to be displayed in the current page.

Create Protocol

Choosing this menu item calls the Protocol Editor Page, initialized with a blank protocol. The Protocol Editor can then be used to create new protocols for the librarian, so that it may communicate with additional instruments. The Protocol Editor is discussed in its own chapter.

[see Protocol Editor chapter]

Modify Protocol

Choosing **Modify Protocol** calls the Protocol Editor Page, in this case containing the protocol for the current library. The Protocol Editor can then be used to modify the protocol for the current library file. Protocols for instruments by the same manufacturer are often similar with slight adjustments. By first loading a library file, and then editing its protocol, other protocols can be created.

Trick: Load a library file from a friend, call **Modify Protocol** from the Edit menu to get to the Protocol Editor Page. From there, call **Save As** from the file menu. *Voila!* You now have a new protocol extracted from your friend's file to be used for receiving data from your synth.

Edit Entry

("E" from the keyboard)

This menu item is used to call a Patch Editor to modify the patch in the current entry in the current page. In order for this to happen correctly, three conditions must be met. 1) There must be a library file displayed in the current page. 2) The protocol for that library file must specify a Patch Editor, and a pathname to that editor. 3) The Patch Editor must be where the protocol says it is (it must be in the disk and directory pointed to by the pathname in the protocol for that file.)

[see File Requester chapter]

[see Protocol Editor chapter]

Example: Using the D-50 Patch Editor.

- ☐ Use the File menu to load the library file “D-50.DemoPatches1.Libr” from the “Libraries” directory of the Music-X examples disk. A file with sixteen D-50 patches should appear in the current page.
- ☐ Select the patch to be edited by clicking on its name in the display panel, or by using the CursorUp and CursorDown keys to move through the list.
- ☐ Choose **Edit Entry** from the Edit menu, or press the “E” key on the Amiga keyboard.
- ☐ Music-X will look in the protocol for the current page to find the pathname for the Patch Editor. In this case, the protocol points to the “Editors” directory on the Music-X Examples disk. Music-X will look for the D-50 Patch Editor and load it.
- ☐ Use the D-50 Patch Editor to modify the current patch. When done, choose **Exit** from the **Action** menu in the patch editor to return to the Librarian Page.

More descriptions of the Patch Editors are available in documentation files on the Music-X Examples disk, and the end of this chapter.

[see Librarian Page / Generic Patch Editor]

Options Menu

The Options menu has one menu item.

Use Original Program #s

This option affects the way program numbers are used in conjunction with the **SEND** button. Read it as “Use Original Program Numbers”.

[see Librarian Page / Mode Panel / SEND]

Normally, when a patch is sent to a synthesizer, it is sent to the program number indicated by the **Program** box in the Mode panel. However, when the **Use Original Program #s** option is turned on (appearing checked), the program number listed next to the page entry is used instead. What this number represents is the original program number that was holding the patch when the patch was first received. It can be useful to send patches to their original locations if preparing the synthesizer to play a performance that includes Program Change events pointing to certain patches (because if the patches are not where expected, the music will sound different).

When the **Use Original Program #s** menu item is off (appearing unchecked), the program number in the **Program** box at the top of the screen is used. (When sending multiple entries using the **ALL** option, that number becomes the base program number for the page.)

♦ Note: In most cases, if the **PREVIEW** button is on, it overrides the **Use Original Program #s** option.

[see Librarian Page / Mode Panel / PREVIEW]

Modules Menu

There are currently no Modules installed in the Modules menu of the Librarian Page. However, the menu is provided for future expansion.

Mode Panel

The top panel of the Librarian Page is called the “Mode Panel” because it contains controls that govern the overall operation of the Librarian Page. It is from this panel that you initiate the sending and receiving of library data. It is also here that you set the optional mode switches governing the handling of program numbers.

- - - Format - - -

At the left end of the Mode panel is an indicator titled **Format**. This is used to show the type of data that the current page holds. Text will be written in the blank area underneath the **Format** title. The Format indicator will show the instrument name followed by the type of data that the page holds. For example, **D-50.Voices**. What is actually displayed here is the name of the protocol being used by the current library. This indicator will change to show the formats for each new page as it is displayed.

SEND

(“S” from the keyboard)

The **SEND** button is used to send library data to a synthesizer or MIDI device. Depending on the status of the other mode switches in the Mode panel, **SEND** will either send the current library entry or all entries. **SEND** is used by clicking on it. While actively sending, the **SEND** button will appear darkened.

Here are some examples of the steps in transferring patch data. Because the Music-X librarian is an open-ended design (being configurable by protocols),

various data types may require different handling. However, the following procedures outlining the basic steps in transferring patch data can be used as guides in most cases.

Example: Previewing a single patch by sending it to a synthesizer's play buffer.

- ☐ Open a library file with **Load** from the File menu. The file must be in single-voice format, designed for your particular synth. (Some libraries can be found on the "Libraries" directory on the Music-X Examples disk.) Once loaded, the file will appear in the current page, with the list of program numbers and patch names.
- ☐ Choose the entry to send by clicking on it or by using the CursorUp and CursorDown keys to move the highlight marker to the desired entry.
- ☐ Turn the **ALL** button off. (We will only be sending one patch.)
- ☐ Turn the **PREVIEW** button on. (This will cause the patch to be sent to the synthesizer's temporary area.)
- ☐ Click the **Channel:** button corresponding to the MIDI channel that your synth is receiving on.
- ☐ Click the **SEND** button. The patch will be sent to the synthesizer. Play the synthesizer to see if you like the sound.
- ☐ If, after previewing the patch, you wish to store the sound in the synthesizer's permanent memory, you must now either do so manually from the front panel of the synthesizer. (Or, with some protocols, and with certain synthesizers, re-sending the patch with the **PREVIEW** button off will write it to the synthesizer's permanent memory.)

Example: Sending a single patch to a synthesizer's permanent memory area.

◆ Note: For this procedure to work, the synthesizer and protocol must both support the writing of single-patch data to permanent memory. Some synths only support the writing of "bulk-dumps" to permanent memory. If the protocol requires "hand-shaking" then the synthesizer's MIDI OUT must also be connected to the Amiga's MIDI IN, and the synthesizer's send channel must be the same as its receive channel.

- ☐ As before, open a library file in single-voice format for your particular synth. The file will appear in the current display panel.
- ☐ Choose the page entry to send by clicking on it or by using the CursorUp and CursorDown keys to move the highlight marker.
- ☐ The **ALL** button should still be off.
- ☐ Turn the **PREVIEW** button off.
- ☐ Set the **Channel** button corresponding to the MIDI channel that your synth is receiving on.
- ☐ Turn off the **Use Original Program #s** option in the Options menu.
- ☐ Set the program number by clicking on the arrows next to the **Program:** box. (This sets the destination program in the synths memory.)
- ☐ An alternative to the two last steps is to turn the **Use Original Program #s** option on in the Options menu. This overrides the number in the program box, and uses the program number listed next to the library entry in the page display.

- ☐ If necessary, manually enable the synthesizer to receive. (Synthesizers, as a safety precaution, often require you to turn off their memory protect functions before they can receive permanent data.)
- ☐ Click the **SEND** button. The patch will be sent to the synthesizer and stored in the synthesizer's permanent memory, using the specified program number.

Another way to send patches to a synth is with a "bulk dump". A bulk dump contains the complete contents of a synthesizer's program memory. Depending on the manufacturer, this is usually 16, 32, 64, or 128 programs. The recommended file extension for protocols that send or receive the complete contents of memory of a synth is ".Bulk". For example, "DX21.Bulk".

Example: Sending bulk data to a synthesizer.

◆ **Note:** For this process to work, the synth must be able to receive data intended for its permanent area and the protocol must be written to support this.

- ☐ Use **Load** from the File menu to locate and load a library file in bulk format for your synth. (Files in bulk format will be appended with the file extension ".Bulk".) The file will appear in the current page. When using this format, each library entry contains a bulk dump for the entire capacity of the instrument.
- ☐ If there are multiple entries in the page, choose the entry to send.
- ☐ Turn the **ALL** button off.
- ☐ Turn the **PREVIEW** button off.

- ☐ The **Program** box is ignored.
- ☐ Set the MIDI channel to the channel that your synth is receiving on. (If handshaking, set your synthesizer's send channel to the same number as well.)
- ☐ If necessary, manually enable the synthesizer to receive bulk data. (Synthesizers often require you to turn off their memory protect functions before they can receive bulk data.)
- ☐ Click the **SEND** button. The entire memory of the synth will be rewritten with the bulk data in the library entry.

RECEIVE

("R" from the keyboard)

The **RECEIVE** button is used to move data from a synthesizer or MIDI device into the current page entry. The type of data to be received is determined by the format of the current page (in other words, by the protocol attached to the current library). Typical formats are: single voices, bulk dumps, performance data, etc. **RECEIVE** is used by clicking on it. When active, the button changes to a darker color.

For Music-X to receive data from a synthesizer or other device, the device's MIDI OUT must be connected to the Amiga's MIDI IN. The simplest way to do this is to temporarily use the cable normally coming from the mother keyboard, since you know one end is already connected to the computer's MIDI IN. Unplug the end of the cable connected to the mother keyboard's MIDI OUT, and plug it in the transmitting device's MIDI OUT. When you are done transferring data, plug the cable back where it belongs. A more elegant (and costly) configuration is to connect all the MIDI OUT ports of the synths in your setup to the MIDI IN ports of a MIDI switcher or MIDI merger box, and to connect the MIDI OUT of the merger to the

computer's MIDI IN. That way every device is constantly available to transmit to the computer as needed, without cables having to be plugged or unplugged.

Receive protocols are of two types: handshaking and non-handshaking. With handshaking protocols, the computer can send a message to the synth that means "please send me your data", after which the synth will respond by doing so. For this to happen, two-way MIDI communication must be possible. In other words, MIDI cables must be connected from the MIDI OUT ports of each to the MIDI IN ports of the other. For non-handshaking protocols, only the MIDI OUT of the device must be connected to the MIDI IN of the Amiga.

Example: Receiving a single patch from the mother keyboard using handshaking.

- ☐ Open a new library by selecting **New** from the File menu. When the file requester appears, choose a single-voice protocol for your keyboard.
- ☐ Set the keyboard's send and receive channels to the same number, and set the librarian's **Channel** buttons to that number as well.
- ☐ Turn off the **PREVIEW** button.
- ☐ Use the **Program:** box to set the program number to download from the synth. (Each program number on the synth can hold a different patch. By choosing a program number, we choose a patch.)
- ☐ Click the **RECEIVE** button.
- ☐ Music-X will ask the synth to send a single patch, specified by the current program number.
- ☐ The synth sends the program. The current library entry receives the program data.

- ☐ To receive additional single patches, select the next blank page entry, change the program number, and click **RECEIVE** again.
- ☐ When done, save your new library using **Save** from the File menu.

You can see in the last example that it's possible to fill up a library with single patches by receiving patches individually. If this is your objective, there is a slightly faster method using the **ALL** button. The **ALL** button tells Music-X to ask for, and receive, 16 separate single patches, one after the other, automatically. The program number and page entry are increased by one with each transfer.

Example: Receiving 16 individual patches using **ALL**.

- ☐ As before, open a new library, formatted for single voices for your synth, and set the keyboard's send and receive channels to the same number as the librarian's **Channel** buttons.
- ☐ Turn off the **PREVIEW** button.
- ☐ Turn on the **ALL** button.
- ☐ Use the **Program:** box to set the base program number to download from the synth.
- ☐ Click the **RECEIVE** button.
- ☐ Music-X asks the synth to send a single patch, specified by the program number.
- ☐ The synth sends the program. The first library entry receives the program data.
- ☐ Music-X increases the program number by one, moves to the next library entry and asks for the next patch.

- ☐ This process continues until 16 patches have been received.
- ☐ When the transfer is done, save your new library file.
- ☐ Any sequential set of 16 programs can be received using **ALL**. If your synthesizer contains more than 16 programs, you can capture the rest by opening another library file, setting the base program number to a higher number and clicking **RECEIVE**. Example: Base program number 17 receives programs 17-32, base program number 33 receives programs 33-48, etc.

If your objective is to receive all the patches currently in your synth, a quicker method to do so is to use a bulk dump.

Example: Receiving bulk data from a synthesizer, using handshaking.

- ☐ The Amiga's MIDI OUT must be connected to the synthesizer's MIDI IN, and the synthesizer's MIDI OUT must be connected to the Amiga's MIDI IN.
- ☐ Choose **New** from the File menu to open a new library. When the file requester appears, use it to select a protocol in bulk format for your synth. The protocols provided with Music-X are found in the Protocols directory in the Music-X program disk. The current page will show the new library with all its entries empty, and the **Format** indicator will show the current protocol.
- ☐ Make sure the keyboard's send and receive channels are the same as the librarian's **Channel** setting.
- ☐ Click the **RECEIVE** button.
- ☐ The **PREVIEW** button is ignored.
- ☐ The **Program** box is ignored.

- ☐ Music-X asks the synth to send the bulk dump for the complete contents of its memory.
- ☐ The synth sends the bulk dump. The current library entry receives the data.
- ☐ Title the data by clicking on the name portion of the page entry to activate the cursor, then typing in a name and pressing the RETURN key.
- ☐ Name and save your new file using **Save** from the File menu.
- ☐ Since the complete contents of the synthesizer were stored in one page entry, and there are 16 free entries in a page, 16 times the memory of the synth may be stored in one library file using bulk transfers.

Certain early MIDI synthesizers cannot do handshaking and must be manipulated from their front panels to get them to transfer data. "Generic" protocols can be used with these synths. The basic process with a generic protocol is to tell Music-X to wait for some data, then go over to the synth, and cause it to send. Whatever the synth sends will be accepted and stored in the current page entry. Two types of generic protocols are provided with Music-X: "Generic.Single", and "Generic.Multiple".

The Generic.Single protocol is used to capture one data message of any type from a synthesizer or device. Some synthesizers send their bulk data as a series of individual messages, each containing one program. The Generic.Multiple protocol can be used with these synthesizers to allow Music-X to capture all the messages without stopping and put them in one library entry.

Example: Receiving bulk data using a generic protocol.

- ☐ Select **New** from the File menu. Then use the file requester to choose a generic protocol and open a new blank page.

- ☐ The **PREVIEW** button is ignored.
- ☐ The **Program** box is ignored.
- ☐ The **Channel** buttons are ignored.
- ☐ Click the **RECEIVE** button. It will become darkened and the librarian will wait for data.

♦ Note: If you decide not to receive data once the **RECEIVE** button is on and waiting, you may turn it off again by clicking it.

- ☐ From the front panel of the synth, initiate bulk data send. (The method for doing this will vary with the instrument; consult the owner's manual.)
- ☐ If you are using a Generic.Single protocol, Music-X will capture all data until it receives an "end of system exclusive" message (the synth sends that to indicate that it's finished).
- ☐ If you are using a Generic.Multiple protocol, manually turn off the **RECEIVE** button to tell Music-X to stop capturing data when the synth is done.
- ☐ The data will then appear in the current entry of the library file, and the **RECEIVE** button will turn off.

Program:

("P" from the keyboard)

To the right of the **RECEIVE** button is the **Program** box. This box is used to set the program number used by the librarian when sending or receiving program data. The program number can be edited by clicking on the small white arrows next to the numeric box and holding down the mouse button. You may also press the "P" key to activate the cursor, and then type in the desired program number with the Amiga's numeric keys.

The number in the **Program** box is overridden by two other options. First, if the **PREVIEW** button is on, any transfers will be made to and from the synthesizer's "temporary area". Second, if the **Use Original Program #s** option is turned on in the Options menu, then during send, the program numbers displayed next to the entries in a page will be used instead of the current program number.

ALL

("A" from the keyboard)

ALL highlights all the entries on a page. While **ALL** is on, **SEND** and **RECEIVE** operate sequentially on all the entries in the list. While active, the **ALL** button will appear darkened.

This works in conjunction with the **Program** setting. If receiving all the entries on the list, the program number represents the base program number for the 16 entries. For example, if you wish to receive the sixteen programs from 1-16 on the synth, set the program number to 1, turn on **ALL**, turn off **PREVIEW**, and click **RECEIVE**.

PREVIEW

("V" from the keyboard, think of "VIEW")

This button turns on and off the preview option used when sending programs to synthesizers or devices with the **SEND** button. The **PREVIEW** button appears darkened when the preview option is turned on.

Most synths use a single-patch buffer in their own memory to store the settings for the current sound they are playing. Some of them call this the "play buffer", or "temporary area". As different patches or programs are called up from the synthesizer's memory, they are loaded into the play buffer so the synth can play it. When using the librarian to send patches to a synth, you may want to try different sounds, to "preview" them, without erasing the settings currently in the synth. This can be done by sending patches one at a time to the synthesizer's play buffer instead of to

its permanent memory. This is what the **PREVIEW** button is for. The **PREVIEW** button is a “flag”, an advisory message, to the protocol. When it is on, it says to the protocol, “send a patch to the synthesizer’s temporary play-buffer area, if you know how”. Not all protocols know how to send to the play buffer (for instance bulk dump protocols). Protocols that don’t know how ignore the **PREVIEW** button.

Examples of this are given under the descriptions of the **SEND** and **RECEIVE** buttons.

Channel:

Below the Mode panel is a panel containing a row of 16 buttons representing the 16 MIDI channels, and identified with the title **Channel**. Protocols use the channel number as an identifier for instruments of identical models used in the same MIDI setup. These buttons will determine the MIDI channel that is used when sending and receiving system exclusive messages to and from a particular instrument.

Always set the channel buttons to the same channel that your synth or device is receiving or sending on. If they are set to some other channel, in most cases, no transfer will occur.

Page Displays

The two panels at the bottom of the Librarian Page are called “pages”, and are used to show the contents of libraries. Up to 10 library files may be resident in memory. Two may be shown at once—one in each page. The **Set Display** option in the Edit menu is used to change which file is currently shown in each page. The pages have identical features. Following are descriptions of those features.

[see Librarian Page / Edit Menu / Set Display]

At the top of each page is the name field. When no library is being shown, the title field reads "No Page". When a new library is created, the name field reads "Untitled.Lib". Once a file has been named (by being saved) the title name field will show the name of the library.

Along the left edge of the panel is the program number column. This column is used to show the original program numbers that the patches were received from. If the entries do not contain individual patch data (they may, for instance, contain bulk data), then this field will not show program numbers. This panel is also used to indicate the current entry. A white highlight bar will be placed in the program number column of the current entry. (The highlight bar also shows which of the two pages is currently active.) When the **All** button is on, the highlight bar will extend to cover the program numbers for all 16 entries, showing that all entries are selected.

In the main body of the panel is the name area. This is where the name for each entry is displayed. When a new library is first created, the names of its entries will all read "empty". When using a single patch protocol, the patch names are copied here during receive. These names may be changed, but they do not change the actual patch name imbedded in the patch. A Patch Editor must be used to do that. When using a bulk dump or other non-patch protocol, the entries may be named anything descriptive. Click in the name area to activate the cursor, then edit the name using the Amiga's keyboard.

To the right of each entry name is a size field. This shows the number of bytes used to store each entry in the Amiga's memory.

Generic Patch Editor

One Patch Editor is always available in Music-X. This is the Generic Patch Editor. The Generic Patch Editor allows you to edit hex data directly, when a regular Patch Editor is not available. In order to effectively edit patch data with the Generic Patch Editor, you must know the structure of the particular bulk data dump being edited.

The Generic Editor appears automatically after an attempt has been made to edit a library entry (by choosing **Edit Entry** from the Edit Menu, or by pressing the “E” key) if the Patch Editor specified in the protocol attached to the current library does not exist or cannot be found. You will first see a prompt that reads **ERROR Couldn't open the file**. Clicking **OK** or pressing the RETURN key causes the prompt to close, and the Generic Patch Editor to appear.

♦ **Note:** If you are sure that you have a Patch Editor on disk but can't seem to get it to load, try this. Use the Protocol Editor to examine the **Patch Editor Name** string for the current protocol. This string is used by Music-X as a pathname when finding the Patch Editor file, and must be correct. Patches received from friends might have custom pathnames specific to your friends' systems.]

[see Librarian Page / Edit Menu / Modify Protocol]

[see Protocol Editor / Name Panel / Patch Editor Name:]

The Generic Editor has two panels. The first is used to hold filename information, and the second panel is used to display and edit the actual patch data.

Three text fields are displayed in the first panel. **File** is used to display the name of the library file from which the current entry being edited is drawn. **Length** is used to show the number of bytes of data stored in the current entry. **Name** is used to display the name of the current entry being edited. These three text fields are for display only, and cannot be edited directly.

The editing panel shows the series of bytes in the current library entry. It is read as a normal page of text; the top line is read first, left to right, then the next line, etc. Each byte is shown as a 2-digit hexadecimal number. Each of the bytes is also numbered according to its position in the series. At the left of each line is the number for the starting byte in each line. Along the top of the list are the hexadecimal numbers 0 through F. By adding these two numbers together one can identify the location of any byte in the file.

A cursor is active in the list. The value of an individual byte may be changed by moving the cursor to it and then either typing in a new value, or pressing the Amiga's plus ("+") and minus ("-") keys to increase and decrease its value. If the file is longer than can fit on the screen, the scroll bar to the left of the panel can be used to display different portions of the file in the window. The window also scrolls automatically with cursor movements. The length of the file cannot be changed.

Along the right edge of the edit panel is a column used to display the ASCII character associated with the value of each byte. This can be handy for quickly locating text portions of a file, such as patch names. Bytes whose ASCII values cannot be typed are displayed as a period (".").

The Generic Patch Editor is exited through its Edit menu. Choose **Exit** or press RightAmiga-"X".



Chapter 13 - Protocol Editor

Overview	362
The Face of the Protocol Editor	365
Title Bar.	366
File Menu	366
Name Panel	367
RECIEVE Panel	369
SEND Panel.	370
SUB-STRINGS Panel.	371
Protocol Language.	372

Protocol Editor

MUSIC-X File: Untitled		EDIT PROTOCOL  	
Protocol Name:			Name Offset: 0000
Normal Program:			Name Length: 00
Preview Program:			
Patch Editor Name:			Char Map:
RECEIVE	Initiate:		
	Confirm:		
	Ack:		
	Data:		
SEND	Initiate:		
	Confirm:		
	Data:		
	Ack:		
SUB-STRINGS	Prefix:		
	Cancel:		
	#1:		
	#2:		

Illustration 13.1 Protocol Editor page

Overview

The Protocol Editor page is accessed from the **Edit** menu of the Librarian Page, and is used to create or modify the protocols used by the Librarian page when sending or receiving data dumps from other MIDI devices. Think of this Page as an extension of the Librarian Page.

[see Librarian Page / Edit Menu / Create Protocol]

[see Librarian Page / Edit Menu / Modify Protocol]

Creating new protocols is not easy. Fortunately, you probably won't need to. In most cases you will be using protocols created by other people, while you spend your time creating music instead. Likely as not, you can take full advantage of Music-X without ever reading this chapter.

Suppose, however, that you have an instrument for which you cannot obtain a protocol. There are several options open to you:

- 1) Hire a MIDI Guru.
- 2) Get a MODEM and call a BBS or online information service, such as PLINK (People-Link) or PAN (Performing Artists Network) or BIX (Byte Information Exchange) to name just a few.
- 3) Find a friendly neighborhood whiz kid.
- 4) Join a User Group; they often have droves of unsuspecting volunteers.
- 5) Write your own protocol.

If you wish to write protocols yourself, you'll need some knowledge of basic MIDI and computer related concepts. For example, many manufacturers publish the specifications for their instruments using the hexadecimal (base 16) or even binary (base 2) numbering systems. It is beyond the scope of this manual to describe technical background such as numbering systems; however the information is readily available in most computer bookstores or in the library.

What is System Exclusive?

When the MIDI standard was being written, it was realized that it would be impossible to think of everything that anyone would possibly want to communicate via MIDI. Therefore a loophole or exception was built into the language. A System Exclusive message (often abbreviated to SysEx; pronounced "sis-ex") can be used to temporarily suspend the rules and let one instrument communicate with another

using codes only they understand. This is like a Roland synthesizer saying “Don’t listen, you guys—this is Roland talk!” That means that even though MIDI is a standard, there is a way for manufacturers to design their own nonstandard communication formats to be used within MIDI. Synth manufacturers have taken advantage of the SysEx loophole to implement data transfers between instruments of the same make and model. The Protocol Editor and Librarian Page are designed to work with the data transfers.

All SysEx messages begin with the MIDI “Start of System Exclusive” code (F0). Following that is the manufacturer’s ID number, and then a series of bytes, the format of which depends on the manufacturer’s ID. Finally, an “End of System Exclusive” (EOX) code is used to terminate the SysEx message.

Modern instruments have lots of different kinds of information in them besides sounds. For example, micro tuning data, performance data, voice parameters, sample data, etc. Each protocol is designed to support one kind of data transfer. It is often the case that a different protocol will be required for each type of information stored in the instrument.

The designer of a protocol is free to use the parameters specified by the Librarian Page **Channel:** buttons, **PREVIEW** button, and **Program** indicator in whatever manner he or she chooses. They don’t necessarily have to be used in the manner in which they are labeled.

The Face of the Protocol Editor

The Protocol Editor Page has four main areas, each containing a number of text boxes. The topmost area contains general information about the instrument, the name of the protocol, which Patch Editor to use if any, etc. The second area contains message strings used in receiving data from the instrument. The third area contains message strings used in sending data to the instrument, and the fourth area consists of “sub-strings” which are messages that are used inside of other messages.

Title Bar

The title bar is found along the top of all Music-X pages to remind one where one is. In the Protocol Editor, the name of the current protocol being edited appears here as well.

File Menu

Protocol files are kept in the protocols directory on the Music-X program disk. If you write or collect more than can fit on your master disk you may wish to move the directory to another disk.

[See Advanced Users / Workbench Tool Types / PROTODIR]

Load...

(RightAmiga-“L” from the keyboard)

This menu item is used to Load a protocol from disk. Choosing **Load...** calls the file requester titled “Load Protocol”. The current protocol name is offered as a default file name by the requester. Loading a protocol from disk overwrites the protocol currently in the editor.

[See File Requester chapter]

Save...

(RightAmiga-“S” from the keyboard)

This menu item is used to Save a protocol to disk. Choosing **Save...** calls the file requester titled “Save Protocol”. The current protocol name is offered as a default file name by the requester. When naming new protocols, the suggested convention

is *InstrumentName.TransferType*. Music-X also saves an icon for the protocol file in the same directory. Icons for protocol files look like a sheet of 1's and 0's.

[See File Requester chapter]

Exit

(RightAmiga-“X” from the keyboard)

This menu item is used to leave the Protocol Editor and go back to the librarian. If you have just created a new protocol, be sure to save it first. The Protocol Editor clears the message strings on exit.

Name Panel

This panel is used to hold protocol information other than the actual message strings. It contains a number of text boxes which are described below.

Protocol Name:

Here's where you name your protocol. This name is used when saving the protocol to disk. The standard (but not mandatory) format for protocol names is: *instrument.datatype*. Example: “CZ-1000.Voices” is the file name for the protocol that sends and receives CZ-1000 individual voice dumps.

Normal Program:

This text box contains one of the substrings substituted for the string variable “PG” in a message string. This substring is used in a message string when the **PREVIEW** button is turned off in the Librarian Page.

Preview Program:

This text box contains the other substring substituted for the string variable “PG” in a message string. This substring is used when the **PREVIEW** button is turned on.

Patch Editor Name:

This is where you attach the appropriate Patch Editor (if there is one) to the current protocol by typing the Editor's pathname. The pathname is stored along with the protocol, imbedded in a library file, when the library file is saved. This allows the Librarian to locate the Patch Editor when the library file is loaded and edited later. If the pathname is wrong, the Librarian will not be able to locate the Patch Editor even if it exists.

[See Librarian Page / Edit Menu / Edit Entry]

Name Offset:

This numeric indicator is used to describe the location of name data within the body of data in a library entry. It is used by the Librarian Page when displaying the names of patches or programs in the individual library entries.

This number represents the number of bytes that must be skipped from the beginning of the library data to find the beginning of the name. A name offset of 0000 means the name starts at the beginning of the data. Various synths imbed the name parameter in different portions of the bulk data dump.

Name Length:

The **Name Length** indicator is used with the **Name Offset** indicator to describe the location and length of name data imbedded within a library entry. It ranges from 0 to 31. A name length of zero can be used in cases where there is no name in the bulk data, as when transferring sample dumps etc.

Char Map:

Read this as "Character Map".

Various manufacturers use different numbers to represent the alphanumeric characters that are considered legal in patch names. Not all use the standard ASCII format as used on most computers. This **Char Map** string gadget can be used to define the

set of legal characters for that synth. If the synth uses the ASCII character set, then leave this field blank.

The contents of the gadget are used to describe, to Music-X, the character set that the instrument uses. This is done by typing the order of letters that it uses.

The ordering of the letters is described by entering a series of letter pairs, separated by commas, each pair representing a range of characters which are in the instrument's character set. For example, if the character "A" is the first character in the instrument's character set, and the letter "Z" is the twenty-sixth character, then the first entry in the **Char Map** string will be "AZ", indicating the letters "A" through "Z". If the twenty-seventh character was the letter "a" (lower case), which then proceeded though "z" (lower case), the next entry would be "az". That would make the total string so far "AZ,az".

When you describe ranges of characters, you can only connect ranges if they can be expressed contiguously in ASCII. Otherwise, the Amiga, which uses ASCII, will not understand. If the ordering of the letters cannot be expressed as a contiguous ASCII range, then you'll need to define one or more ranges whose widths are one character wide. This is done by delimiting such ranges with the same character (as in "AA").

You can also enter characters with a 2-digit hexadecimal value, by proceeding them with a backslash ("\"). For example, "\20", which is the ASCII value for a space character. This is handy for entering characters like the comma (",") which might be misinterpreted.

RECEIVE Panel

The **RECEIVE** panel is used to define the set of incoming and outgoing message strings that will be used during transfer when the **RECEIVE** button is clicked in the Librarian Page. It contains four string gadgets that can be edited in the normal fashion. All numbers in the string panels are in hexadecimal. This is the adopted format because most instrument's System Exclusive formats are published in hex.

Following are short descriptions of the strings. They are described more fully in the “Music-X Protocol Language” section of this chapter.

Initiate:

This string describes the request to receive, sent by Music-X.

Confirm:

This string describes the confirmation message sent by the synth to indicate that it is ready to send data.

Ack:

Read this as “Acknowledge.”.

This string describes the acknowledgement message sent by Music-X that it is ready to receive data.

Data:

This string describes the format of the data that the synth will be sending.

SEND Panel

The **SEND** panel is used to define the set of incoming and outgoing message strings that will be used during transfer when the **SEND** button is clicked in the Librarian Page.

Initiate:

This is the request to send, sent by Music-X.

Confirm:

This is the “OK, ready to receive” confirmation that the synth sends (if any).

Data:

This is the format for the data that Music-X will be sending.

Ack:

This is the acknowledgement that the synth sends to say that it is ready to receive more data.

SUB-STRINGS Panel

Sometimes there are strings that need to be used often in the **RECEIVE** or **SEND** panels. These message fragments can be written into the **SUB-STRINGS** panel, and then be represented in the other message strings as abbreviations.

Prefix:

The abbreviation for Prefix is "PR". Anytime "PR" appears in a send or receive string, the prefix string is substituted. A Prefix string usually consists of a "Beginning of SysEx" byte (F0), then a manufacturer's ID.

Cancel:

For those synthesizers which understand such things, this substring is used to hold the Cancel string which is sent when Music-X wants to halt the transfer process, perhaps as a result of a user abort. This string cannot be imbedded in any other message strings; it is used automatically when needed.

#1:

This is user string number one. Its contents are substituted for the abbreviation "#1" when used in the other panels.

#2:

This is user string number two. Its abbreviation is "#2".

Protocol Language

Technically speaking, a Music-X protocol is a description of the messages that need to be sent to an instrument to communicate with it. This sub-chapter will explain the construction of those messages using the protocol language built into Music-X.

Protocols, Messages and Strings.

Each protocol consists of a number of “messages”. Each message is a string of codes to be either sent to or received from the instrument. You create a protocol by creating the correct message strings for that instrument, and typing them into the Protocol Editor. The message strings are lists of numbers or mathematical expressions, separated by commas. The numbers must be in hexadecimal (since that’s the way synthesizer manufacturers print out their code descriptions). Once typed in, Music-X can use this information to communicate with the instrument.

Finding out about MIDI formats.

Most manufacturers of MIDI equipment publish the system exclusive codes needed to communicate with their instruments. Some manufacturers publish these codes in the back of the manual, or in a separate book that comes with the instrument. Others require that you write to them directly to obtain a copy of the system exclusive format.

Sending and Receiving.

Communicating with some synthesizers is very simple. When you want to send them a message, you simply send it, and when you want to receive a message, you send a special message to the synthesizer saying “please send me some data”, and the synthesizer sends you the data, which you receive and store away.

However, most instruments are more complex than that. Usually there is some sort of “handshaking”. Handshaking is like good manners, in that there is a lot of “please

and thank-you” going on. Here is an example of handshaking between the computer and a synthesizer:

Computer: Please send a message.

Synth: Are you ready for the message?

Computer: Yes, send the first part.

Synth: Here it is. <data>

Computer: please send the next part.

Synth: Here it is. <data>

Computer: Please send the next part.

(repeat previous two steps for a while)

Synth: Here it is, this is the last part. <data>

Computer: Thank you very much.

Synth: You’re welcome.

Now, that was a somewhat anthropomorphized version of the way a protocol is used. In actuality, it would look something like this:

Computer: F0 44 12 17 19 F7

Synth: F0 44 13 19 00 00 18 F7

(And so forth. You get the idea.)

Now, not all instruments go to this extreme. Many of the messages used in the above example are often absent on a particular synth. Other synths have special messages like “I have a problem, stop sending messages for a while,” or “Something’s wrong, cancel the transfer,” or “Eh? I didn’t get that last message quite right.”

The message strings are used to tell the computer how to send and receive messages of this type. Because of handshaking, each transaction between the computer and the synthesizer will often consist of a number of messages traveling in both

directions. For example, when sending data to the synthesizer, the computer will actually receive as well as send messages, although the important data will be in the messages that are sent. It will be necessary for you to describe correctly all of the incoming and outgoing messages, for both types of transactions.

Music-X protocols are based on a “dialogue model”. That is, as in a conversation; each person takes turns talking with the other person. Here is the overall structure of a transaction to RECEIVE data:

1. Music-X requests data.
2. Synthesizer Acknowledges Music-X.
3. Music-X indicates that it is ready to accept data.
4. Synth sends some data.

(Steps 3 and 4 are repeated until all data is received).

These 4 steps are labeled Initiate, Confirm, Ack (Acknowledge), Data.

Any one of the messages may be omitted, although omitting the last one is not meaningful. For example, if message number 3 (Confirm) is omitted, then it is assumed that the synthesizer will continue to send data without waiting for a confirmation. If message number 2 (Ack) is omitted, then it is assumed that there is no acknowledgement sent from the synthesizer; that the first thing it will send will be a data message. If message 1 is omitted, then it usually means that the synthesizer requires “manual MIDI control”. That is, you actually have to press a button on its front panel to get it to send the data, and there is no actual message that can be sent from the computer to get it to send data.

◆ Note: There is another way that the four messages can be used. Some synthesizers send their data as two separate messages which are different. This is often the case when an instrument comes out which is an advanced version of an older model—one message is sent which represents the old format information, and the second message is additional information which applies only to the new model.

There is a way (explained below), whereby you can use the Ack message as an additional Data message which is different from the regular data message.

The overall structure of a transaction to SEND data to the synthesizer is slightly different:

1. Music-X indicates that it will send data.
2. Synth responds that it is ready.
3. Music-X sends some data.
4. Synth indicates that it got the data OK.

(Steps 3 and 4 are repeated until all data is sent).

Like before, messages #1 and #3 are outgoing, while messages #2 and #4 are incoming, but in this case the bulk of the actual data is in #3 instead of #4. These 4 messages are named "Init", "Confirm", "Data", and "Ack". ("Data" and "Ack" have exchanged places).

Like the receive transaction, any of the messages may be omitted. Omitting the first two messages is often done in cases where the synthesizer does not handshake, but simply expects the data to be sent. Omitting the last message is often done in cases where the data is sent as a single message, or when no confirmation of individual data blocks is needed.

Limitations of System Exclusive Messages.

Because of the way MIDI is structured, each system exclusive message must start with the hexadecimal code "F0" meaning "start of system exclusive message". Each code, or byte, in the message following that must consist of a series of eight bits, and the most significant bit must be zero. Any byte sent with a most significant bit that is not zero (except Real-Time messages such as Active Sensing) will automatically terminate the system exclusive mode. Due to this fact, all of the bytes within the body of the message must have values which are between 0 and 127. (There are 7 bits available, and each bit can be on or off, which makes 128

combinations of ons and offs. Since we include zero as one of the numbers, we have to leave out the number 128. So, 0-127).

Unfortunately, many instruments (for example, samplers) have parameters or data values that are larger than 127. These instruments solve the problem by breaking up the large number into two smaller numbers, each of which has less bits than the large one. The way the numbers are broken up differs for different instruments. You'll need to be aware of this to understand some of the explanations that are to follow.

The LookOut Utility

A program called "LookOut" has been included on the Music-X Utilities disk. It will help you to develop new protocols. What LookOut does is open up a window which prints out error messages and status messages when a protocol is activated, as well as showing the actual bytes that were sent and received. You can use LookOut as a debugger to correct errors in your protocols.

◆ Note: Don't expect to get a protocol correct on the first try — nobody has yet.

Forming a message string.

A message string consists of fields separated by commas. Commas may only be used as separators and may not occur within parenthesized expressions. Spaces between the fields are ignored. Each field is actually a command to Music-X, and can consist of either a hexadecimal number, a mathematical expression, or a special command. Fields consisting of just a hexadecimal number simply mean "Send this number." For example:

F0,44,12,45

This message string simply means that the message consists of the code "F0", followed by the code "44", followed by "12", followed by "45". If the message was an outgoing one, the computer would send those codes to the synthesizer. If it was an incoming message, then instead of sending those codes, the computer would know to expect them.

It is also possible to specify codes that are longer than one byte by placing the codes in parentheses. For example, the field

(0F0F.3)

means send the hexadecimal number “0F0F” as a three-byte code. The “3” is a “length specifier” — it indicates the length, in bytes, of the field in the message string. All length specifiers must be preceded by a period, and must be part of a field surrounded by parentheses. Length specifiers can range from 1 to 5.

Because of the way the Protocol Language works, all multi-byte messages are broken up into chunks of seven rather than eight bits, except for the first byte, which is sent as a full eight bits. So, the bytes “0F0F”, which, in binary, are actually “0000111100001111”, are sent as:

“00000000” (8 bits)

“00011110” (7 bits + leading zero bit)

“00001111” (7 bits + leading zero bit)

The reason for this is that all of the byte codes in a system exclusive message, except for the first one, must have the first bit as zero. So that only leaves seven bits. This system of encoding works well with popular synthesizer formats.

ASCII Strings

You can also put an ASCII string into a field, in which case the string is put into the message, character by character. The string must be separated by double quotes (“”).

An ASCII character surrounded by single quotes, such as 'A' means to use the numeric value of that ASCII character (just like in the C language). Singly quoted letters may be used in expressions, and may have length specifiers.

Examples:

F0,44,"AAAA",10

produces the codes:

F0 44 41 41 41 41 10

(The ASCII code for the letter "A" is 41 in hexadecimal).

('A'.3) — means "take the value 41 and insert it into the message for 3 bytes".

Expressions

Message fields can also contain mathematical expressions (in parentheses). For example, instead of including a "2" in the message, you could include the expression "(1+1)". Expressions can also have a length specifier, as in: "(5+9*3.5)" which means, "Take 5, add 9, multiply by 3, and put the result into the next 5 bytes of the message."

All of the terms in an expression must be integers, and evaluation is in left to right order, (except for portions within parentheses, which are evaluated first (parentheses can be nested (to any level))). The operators supported are:

+	Add
-	Subtract (or Negative)
*	Multiply
/	Divide
%	Modulus
!	Boolean NOT
&	Boolean AND
^	Boolean Exclusive OR
	Boolean OR
<<	Logical Bit Shift Left

- >> Logical Bit Shift Right
- () Parenthesized expression
- => Assignment — Variables can be assigned values, as in
(A+1 => A) where the value of A+1 is assigned to A.

Wild Card character

Sometimes you don't care about a particular incoming code from the synthesizer, or don't know what that code will be. There is a wildcard command that you can use instead of a number. The command is "##". You can use it by itself, in which case it causes one byte of the message to be ignored, or in parentheses with a length specifier, in which case the length specifier indicates how many bytes are to be ignored. For example:

F0,41,(##.3),42

This tells Music-X to expect two bytes (F0 and 41), then ignore the next three bytes, then expect the byte "42". Wildcards can only be used on incoming messages; otherwise, a protocol error will result.

Numeric Variables

A numeric variable is a single letter used to represent a numeric value. The value represented by a variable may vary (thus "variable"). Variables can be used within message strings to allow the string to change based on some external circumstance. For example, the message string used by Music-X to initiate a program dump by a synthesizer could change depending upon which program you wished to download. Whenever a variable is used in a message string, the current value of that variable is substituted into the message string. For example:

12,33,n,02

In the above example, the third byte of the message will be whatever the variable "n" is currently set to. Variables can also be used with length specifiers, as in "(n.4)".

All of these variables have pre-defined meanings. You cannot create your own variables, as you would in a programming language. The meanings of the variables are as follows:

“K” or “k” — Checksum

Many instruments use checksumming as a way of insuring that the data has been transferred correctly. A checksum is created by taking all or part of a message and adding up those codes numerically to form a sum. This sum is then sent along with the message. When the message is received, the receiver adds up the codes, and checks to see if his sum matches the checksum, in which case the receiver knows that message got through OK. This can be important if the messages are very long, because a slight power fluctuation or static shock to the computer could cause a byte to be received incorrectly.

To use checksumming, you need to put braces (the “{” and “}”, or “curly bracket” characters) around the part of the message which is to be summed. Multiple fields may be enclosed in braces and checksummed. Example:

12,34,{02,n,55}

This causes the third through fifth byte of the message to be added together and placed into the variable “K”.

Many synthesizers use a negative checksum, which means that the checksum value sent is the negative sum of the message bytes. This has the advantage that the message bytes and the checksum added together equals zero. To form a checksum like this, use an expression (-K&7f.1) which means “Take the sum “K”, negate it, logical AND it with 7F (which turns off the topmost bit, necessary for MIDI system exclusive data), and insert it into the message string for 1 byte.

“J” or “j” — Special Checksum

This variable functions the same as “K”, except that the checksum is calculated by using boolean XOR rather than addition.

“M” or “m” — Total Blocks Sent

This variable starts at zero and is increased by 1 for each block or section of data that is sent to the instrument.

“N” or “n” — Channel Number

This variable is set to the currently selected channel in the Librarian Page. (Actually, it's one less than that, as channel numbers actually start at zero, but are labeled starting from one).

“O” or “o” — Patch offset, only in send

This variable is only used in special protocols that work in cooperation with certain patch editors. It is used when a patch editor wants to send a “partial update”. This allows a patch editor to change some of the data in a patch, and only send to the instrument the part that was changed. The “O” variable indicates the start of the changed section. For example, if the patch data was 488 bytes long, and the bytes 52 through 69 had been changed, then “O” would be set to 52.

“P” or “p” — Program Number

This variable is set to one less than the current value of the **Program:** indicator in the Librarian Page. It ranges from 000 to 127.

If **Use Original Program #s** is selected, in the Librarian Page / Options menu, and we are sending data, the variable “P” will actually be set to the original program number that the patch was received from, as shown in the library file.

“Q” or “q” — Send Data Size

This variable indicates the size of the data block that is to be sent to the instrument. Normally, this will be the full size of the patch, but it could be less for “partial updates”. This variable is only used to inform synthesizers that need to be told how much data to receive.

“R” or “r” — Total Bytes Received

The total number of data bytes received so far in the transaction (after packing if used with “Y” data).

“S” or “s” — Total Bytes Sent

The total number of data bytes sent so far in the transaction (after packing).

“U” or “u” — Total Blocks Received

This variable starts at zero and is increased by 1 each time the protocol goes through the Ack-message/Data-message cycle.

“V” or “v” — Value Last Received

This variable is set to the value of the last data byte received.

String Variables

Another type of variable that can be used in a protocol specification is a string variable. Each string variable corresponds to one of the substrings in the bottom panel of the Protocol Editor. For example, the string variable “PR” stands for “Prefix”. Whenever a message field contains the letters “PR”, the contents of the **Prefix**: message text box will be substituted in its place. Message strings can get very long in some cases, as well as very repetitive, so it’s handy to be able to define commonly used parts of messages. There are four string variables. They are as follows:

“PG” — Program string

This variable will be substituted with one of two strings, depending upon whether the **PREVIEW** button in the Librarian Page is set. If the **PREVIEW**

button is set, then the message string labeled **Preview Program:** will be used; otherwise, the string labeled **Normal Program:** will be used. This will allow the message protocol to vary based on the setting of the **PREVIEW** button. In many instruments, the “temporary area” is accessed by altering the program number sent to the synthesizer.

“PR” — Prefix

This variable will be substituted with the contents of the **Prefix** message string. In almost all synths, the first few bytes of every message that the synth will send or receive are the same as the first few bytes of all of its other messages. So to save typing, you can put those bytes into the prefix string.

“#1” — User string 1

This variable will be substituted with the contents of the **#1:** message string.

“#2” — User string 2

This variable will be substituted with the contents of the **#2:** message string.

String variables can be used by themselves in a field, or as a component of an expression if the resulting string is properly formed. For example, you could use “(PR*6)” if PR was “04” but not if PR was “04,03” (because you can’t have a comma within a parenthesized expression).

Data Variables

Data variables represent data that is to be sent or received. When received, this is the actual data that will be stored in the library entry; all other parts of the system exclusive message (header, checksum, Ack, etc.) are discarded after being used. When being sent, this data is read from the library entry, and inserted into the transfer at the correct point. Data variables are specified by the letter “X” or “Y”. Each variable must be used in an expression followed by a length specifier, and surrounded by parentheses, and indicates how many bytes of data are to be transferred in each block. In addition to a length specifier, the Data variables may have

an “offset” specifier, and a “total” specifier. The format for Data Transfer message fields can be as follows (including parentheses):

(Data.Length)

(Data.Length:Total)

(Data.Length.Offset)

(Data.Length.Offset:Total)

Data

There are currently two forms of Data variables. Each can be used for sending or receiving.

The “X” command means the data is transferred normally, i.e. not encoded in any way. (This requires that the data is already in normal format with the leading bit zeroed.)

The “Y” command means the data is transferred in “nybbleized” form. Each byte of the data is sent or received as two bytes. The first byte contains the rightmost four bits of the original data byte, and the second byte contains the leftmost four bits. So, a command of (Y.100) will actually cause 200 bytes to be transferred, however these will be packed back into 100 bytes after transfer. Length specifiers, when used with “Y” data, always specify the number of bytes in packed form.

Length

The Length specifier follows the command field and is separated from it by a period. It indicates the size of the data block to be transferred. If there is only one block then it indicates the total size of the data.

A length specifier of zero indicates a special case that causes the Librarian to ignore EOX (F7H, End of System Exclusive) and continue to capture

everything until the receive button is turned off. For example:

(X.0)

Total

The Total specifier can be used during receive only. It follows either the Length specifier or the Offset specifier. It must be last in the field and is separated from the previous specifier by a colon (":"). It indicates the total number of bytes of all data blocks to be received. If using nybbleized data, it specifies the total number of bytes in packed form. Music-X will continue to send and receive messages until the total amount of data transferred is equal to or greater than this number.

Offset

There are some cases when data needs to be sent to an instrument in a different order than it was received. The Offset specifier is a way of implementing that.

The Offset specifier is only used when sending data. It must follow the Length specifier and is separated from it by a period. The Offset specifier can be positive or negative, and is used to read data starting from any point in the library file.

Normally, Music-X sends out data sections sequentially, starting from the beginning of the library file. For example, if the patch data record was 100 bytes long, and you instructed Music-X to send 25 bytes (say using (X.25)), the first time the message was transmitted Music-X would send bytes 1-25 of the data. The second time it would send bytes 26-50, and so on. However, if you specified an offset of 10, as in (X.25.10), it would send bytes 11-35 the first time, bytes 36-60 the second time, etc.

The way this works is, there is a “counter” that counts the number of bytes of library data sent. This counter is identical to the “S” variable. It starts at zero and increases by one with every byte. This counter is normally also used to point to the next byte of library data that is to be sent. When an Offset specifier is used after a Data Variable, the Offset and the counter are added together to calculate which byte in the library file to send.

◆ Note: Refer to the TX-81Z.Voices protocol for an example of this.

Special Commands

In addition to the above, there are a number of miscellaneous special commands:

“W” — Wait

Tells Music-X to wait for a specified time. The length specifier is used as a delay time, and is in 50ths of a second. For example:

(W.50)

This field causes Music-X to wait for one second before continuing with the message.

“*” — Sentinel**

Sometimes it is not possible to tell exactly how long an incoming message is going to be, as some instruments send messages of varying length. For this reason, the “sentinel” command has been included. If you know that a particular byte value is always going to be at the end of a message, you can use the sentinel command to inform Music-X that it can use that value to

detect when the message ends. To use sentinel, place the "***" immediately before the value to be used. For example:

44,56,73,99,**F7

This example tells Music-X that the first "F7" received will indicate the end of the message. Sentinel values are usually the last byte of a message.

"@@" — Special Sentinel

This is used to indicate the last data block in the transfer. To use sentinel, place the "@@" immediately before the value to be used. When this value is detected during receive, the protocol breaks out of the Ack-message/Data-message cycle.

When this value is to be used during send, the value is sent with the last block at the specified point.

"FIN" — Send Final Message

Some synthesizers require that when receiving data from them, a final "Ack" message be sent after all data has been received. The FIN command, if included in a message, will enable this behavior. The "FIN" command can be anywhere in the "Ack" message.

Chapter 14 - Tutorials

Amiga Basics.	389
MIDI Overview.	395
Setting Up a Metronome Track.	396

Tutorials

Amiga Basics

First, refer to your Amiga owner's manual for directions on how to set up the hardware and plug everything in.

The Amiga is equipped with a generalized user interface called "Intuition". This includes the basic things you use to control the computer, i.e. the mouse and mouse pointer, screens, menus, windows, gadgets etc. Intuition is so named because you can quickly get an intuitive feel for how things work.

You'll find that most software for the Amiga makes use of these Intuition features. Music-X, however, actually improves on them. This chapter will explain the basic use of normal Intuition features as well as the augmented Music-X features.

Using Menus

Menus are used on the Amiga to offer a list of choices, thus the name "menus." Menus on the Amiga are of the "pull-down" type, which means that they don't appear on the screen until you pull them down from the "Menu Bar" at the top of the screen. Here's a general process for using menus.

- ☐ Press and hold the RIGHT mouse button.
- ☐ See the highlighted Menu Bar at the top of the screen, containing the titles of available menus.
- ☐ While holding the mouse button, move the mouse pointer up to a menu title.

- ☐ See the menu pull down with a list of “items”.
- ☐ While still holding the mouse button, pull the mouse pointer down through the list.
- ☐ See the different items become highlighted.
- ☐ Stop at a selected item.
- ☐ While the item you want is highlighted, release the RIGHT mouse button.
- ☐ The menu will disappear, the menu bar will disappear, and your selection will be activated.

Certain conventions are followed in Music-X when displaying the text for menu items. Markings are given to indicate the type of item being shown. For example:

Item...

Three dots after a menu item means that the item calls a window or module.

Item - - ->

A dashed line with an arrow after a menu item means that it has a “pop out” menu attached (also called a “submenu”). Pop out menus “pop out” to the right of the parent menu when the pointer is on the menu item. Move the pointer into the pop out menu and choose items in the normal way; by releasing the mouse button when the item you want is highlighted.

√ Item

A check before the item means this item is currently active. Most checked items are turned on and off by selecting the same item again. Some checked items are grouped together and are mutually exclusive. Only one of these items may be checked at a time. Selecting one turns off another.

Using Buttons

Buttons are another type of control used on the Amiga and in Music-X. Buttons are commonly used to initiate actions. Buttons usually appear as one or two words surrounded by a border, although there is a smaller variety that appears as an oval-shaped “mini-button” with the button text to the right of the button proper.

Most buttons in Music-X can be activated in two ways: With the mouse or with the keyboard.

To activate a button with the mouse, place the mouse pointer within the bordered area. Press the left mouse button. This is referred to as “clicking” on the button. The button will light up, and the specified action will take place.

To use a button with the keyboard, you must know which key is the “key equivalent” of that button. This information is listed with the description of each button in the manual, and in an appendix as well. Simply press the key which corresponds to that button, and Music-X will react as if that button had been clicked with the mouse.

Some buttons will only stay lit for as long as you hold down the mouse button. Others, when clicked, will turn on if they are off, and off if they are on, like a retractable ball-point pen. Some buttons, when activated, will stay on, and cause other buttons to turn off. This often appears in cases where only one of a number of options is usable at any given time. Another type button continuously performs its function for as long as you hold it down, like the fast forward button on a tape recorder.

Using Requesters.

Sometimes clicking a button or choosing a menu item will cause a “requester” to open, “requesting” additional input from the user or in some cases containing controls to initiate actions themselves. A requester appears as a bordered rectangle, containing buttons and other controls, in front of the background display, covering part of it. While the requester is active, all of the controls on the background screen will be disabled, and you will not be able to access them until the requester is closed. Most requesters will have some of the same features.

CANCEL

Wherever a **CANCEL** button appears, it always provides a way to back out of a situation safely. It’s usually offered as an option in cases where the overwriting of previous data is a likelihood. It will return you to whatever page you came from with everything in the same unchanged state.

OK

This always means, “Go ahead and do the function I’ve started.” Requiring the **OK** button to be clicked is usually imposed as an extra safety step, whenever the overwriting of data is imminent, to prevent accidental erasure.

Dragging

Certain on-screen objects can be moved around on the screen directly with the mouse pointer. To drag an object, move the pointer to the object, press and hold the left mouse button. Moving the mouse pointer while holding the mouse button will “drag” the object across the screen. Sliders can be dragged in a single dimension, but other objects, such as the icons on the Workbench, can be dragged in two dimensions.

Proportional Sliders

Sliders appear as long rectangles with a variable sized “drag bar” inside them.

Proportional sliders are used in Music-X for scrolling through lists or other graphic displays. The size of the drag bar is proportional to the amount of data that is visible in relation to how much is currently off screen.

- ☐ Move the mouse pointer over the drag bar in a slider.
- ☐ Press and hold the LEFT mouse button. The drag bar will change color to show that it has been grabbed.
- ☐ While holding the LEFT mouse button, move the mouse pointer along the slider.
- ☐ Notice that the drag bar is pulled along.
- ☐ Continue to drag the slider until the data that you wish to see comes into view.
- ☐ Release the LEFT mouse button.

You can also adjust the display position by clicking inside the slider’s field of travel, on either side of the drag bar. This will move the drag bar by an amount equal to its width.

Some sliders, in addition to a drag bar, have arrow-shaped gadgets on either end. Clicking on an arrow gadget moves the drag bar in that direction by the minimal increment.

Music-X sliders

Music-X employs an additional, slightly different type of slider, which is used to adjust continuously variable parameters such as tempo. The drag bar is not variable sized, and can display a numerical value. Moving the drag bar changes the value in real time.

Coarse changes in the slider's value are made by grabbing and dragging the slider. Fine changes are made by clicking, or clicking and holding down the mouse button, on either side of the number bar (in the slider's field of travel) in order to increase and decrease its value.

Screens

A screen is a draggable "page" which programs use to display their user interface.

In some systems, a "Screen" simply means the display area of your monitor. The Amiga, however, supports multiple overlapping screens, so that many programs can be running simultaneously, each with its own screen. Each program's screen can have colors, resolution, and windows which are separate and different from any other program's screen.

Dragging Screens

Screens can be moved with the mouse, by dragging the title bar of the screen. This will expose any underlying screens that might happen to be active at the time. For example, if Music-X is running, and Workbench is active, dragging down the Music-X screen will expose the Workbench screen. This will allow you to use the Workbench even while Music-X is playing a song.

Screens can only be dragged up or down, not side to side.

Depth arrangement gadgets

Most screens will have "depth arrangement gadgets", also known as "front and back gadgets". They are found at the upper right corner of the screen. Clicking on these will either hide the screen behind all other screens, or bring the screen in front of all other screens.

There are also two standard Amiga keyboard commands that affect screens similarly. LeftAmiga-"N" from any screen will bring the Workbench screen to the front, and LeftAmiga-"M" will put the Workbench screen to the back, which has the effect of returning you to the screen from which you just came.

MIDI Overview

The following subchapter presents some perspectives about MIDI, and explains some basic concepts of how it works.

A MIDI transmission is a series of “Messages”, each message consisting of a string of numbers. These numbers are not sounds in any way, but rather commands to play sounds.

MIDI is “serial”, meaning everything is sent one bit at a time (albeit very rapidly). Bits add up to bytes and bytes add up to MIDI messages.

There are various kinds of MIDI messages, the most common being Note-On and Note-Off. All this does is transmit a code that says, “Play your key number such-and-such.”

A keyboard is played by pressing the keys. In an acoustic piano, pressing the key causes the hammer and felt to strike a string. In an electronic keyboard or synthesizer, pressing a key closes a switch.

In a modern synthesizer, each key press is monitored by an on-board computer. When a switch is closed, the computer causes a note to be played. That means that pressing a key is not making the sound, rather it is telling the on-board computer which sound to make, and it then does, rather like internal remote control.

In a MIDI synthesizer, all the keys are numbered. Each time a key is pressed, the number for that key is sent out the MIDI line. Another MIDI synthesizer at the other end of the cable can receive that number, and cause a note to be played—the same note that you played on the “mother keyboard”. This is external remote control.

Music-X records MIDI key presses, similar to a player piano, except that the paper rolls have been replaced by digital memory.

MIDI supports 16 “channels”. What this means is that you can send 16 different information streams on one MIDI line. The effect is to have up to 16 different synths or other MIDI instruments playing different things at the same time. In reality this is an illusion. There is only one line and all devices receive the same information. Then why don’t they all play the same thing? The way it’s done is to attach a number from 0 to 15 to every MIDI event. Then each synth is also set to a number from 0 to 15 (although you see it as 1 to 16). Each synth sorts through all the events coming in on the MIDI line and responds only to the events tagged with its own channel number.

Setting up a Metronome Track

The metronome in Music-X is user-written. That is, there is really no “metronome” in Music-X. Instead the user may build a metronome-like sequence in any time signature he or she wants. The stock metronome that comes with Music-X in the “Default.Perf” file is actually a 1-measure sequence, that plays four quarter notes indefinitely. The “Default.Perf” also contains one Amiga Sample, that of a metronome click sound. The metronome sequence is set to play the Amiga Sample; this is how the metronome is created.

The metronome can be completely rewritten by the user, even given another sound, and saved back in the “Default.Perf” to be loaded automatically each time Music-X is started.

To build a metronome sequence in any time signature:

- ☐ Use the controls in the Time Signature window in the Sequencer Page to set the desired Time Signature.
- ☐ Select an empty sequence in the Sequence List.
- ☐ Click **EDIT** to edit the current sequence using the Bar Editor.

- ☐ Use **Add** to place notes on each beat in the measure (or some more complex rhythm if you wish).
- ☐ Use **Add** to add a Set Repeats event.
- ☐ Move the slider titled **RPT:** to increase the amount of repeats to 100 (open repeats).
- ☐ Exit the editor using **Exit** in the File menu, and choose **STORE** from the safety prompt.

Chapter 15 - Advanced Users

Editing Tricks.	399
Song Assembly.	403
Workbench Tool Types.	410
Installing Modules	416
File Conversion	419

Advanced Users

Editing Tricks

The features in the Editor Pages are general purpose tools, with more than one application. By combining tools in “processes” (i.e. multi-step operations) various editing effects can be achieved. Here are a few ideas for some processes.

Quick Transposition

Mark all or part of the sequence, then use the CursorUp or CursorDown key. Holding SHIFT at the same time will transpose by octaves.

Pasting between sequences

The Clip will hold its contents until new contents are put into it. Exiting the editors, or bringing new sequences into the editors, will not cause it to lose its contents. This means the Clip can be used to exchange blocks between sequences. Clip some events out of a sequence, then without leaving the editor, **Select...** (from the File menu) another sequence and **Paste** the Clip into it.

Extracting A Drum Pattern

To glean a short drum pattern from a long improvised drum sequence, **Mark** the best measure and copy it into the clip. Then **Select** a new sequence, and move the End line to the beginning. Now **Insert** the Clip at the beginning and add a Set Repeats event.

Rhythmic Offsets

Offsetting is when sequenced parts are shifted slightly in time to make them start early, or start late against the clock. This can be used to give characteristic effects to individual parts (such as sounding “urgent” or “laid back”), or for compensating for

the attack times of sampled sounds. Offsetting melodic fragments by larger amounts of time can be used as a compositional tool. There are a few methods for offsetting in the Bar Editor.

One method moves only currently displayed events, which can be handy or not, depending on your purpose. Use **Select** or **Select All**, turn **Snap** off, and move everything by clocks with the SHIFT-CursorLeft and SHIFT-CursorRight keys.

Another method is to **Zoom-** way out, **Mark** the entire sequence, and **Cut** everything into the Clip. Now turn **Snap** off, **Zoom+** back in and **Paste** the Clip at the desired clock while watching the **T=** and **D=** indicators.

If you are moving everything later in time, you can **Insert** a Note event with the same duration as the desired offset at the beginning of the sequence and then **Remove** that event.

If you are moving everything earlier in time, you can turn **Snap** off and then **Mark** and **Delete** a region of time equal to the desired offset.

You can also offset a sequence by triggering it with a Play Sequence event from another sequence.

MIDI Doubling

MIDI Doubling is when two MIDI Note On messages (and Note Off messages) are sent for each note in a sequence. This causes the synthesizer to double the amount of voices applied to each note, giving it a thicker sound. This effect can be accomplished from the Sequencer Page by Merging a sequence with itself. A less memory consumptive method is to turn the sequence off and trigger it twice from another sequence using double Play Sequence events.

MIDI Delay

MIDI Delay is like MIDI Doubling, but with some time delay between the original and doubled notes. This emulates the effects of a delay unit type signal processor.

This can be done by Copying and Pasting the notes to be delayed at some time later than the original notes. By then lowering the velocity of the pasted notes, you can get an echo effect. One might even consider transposing the pasted notes by some interval. This method uses extra memory but is handy for smaller sections.

To effect MIDI Delay on an entire sequence, trigger the same sequence twice using Play Sequence events starting at slightly different times.

Random MIDI Delay

You can get a nice variable width “flamming” effect by first Recording a live instrument part and Storing it to a sequence, then Copying the sequence and Quantizing the copy, then playing the two sequences together.

MIDI volume effects

MIDI Volume (controller 7) can be used to achieve various effects. Your synth needs to be able to respond to controller 7 for these techniques to work.

A MIDI fade-in can be created by Adding to a sequence a series of Control Change events set to controller 7 and increasing in value from zero to some audible level. This must be done for every MIDI channel that will be used. Fade-out would be created in the opposite order.

A sforzando is when a horn section attacks a voicing loudly, then instantly drops down to almost no volume and slowly increases in loudness through the rest of the duration. This or any volume envelope can be mapped over a chord by sending MIDI volume events during the middle of the notes. An example of this can be found in the Performances directory on the Music-X examples disk titled “sfz.Perf”.

Rearticulation

To make a part sound more “heavy handed”, slightly increase the velocity of all the notes and lengthen their durations a bit. To do this, use **Select All**, then use the Velocity Scaling module. While the notes are still selected, turn off **Snap** and press SHIFT-CursorRight a few times to extend their durations.

To make the part “lighter”, decrease the overall velocity and shorten the durations of the notes.

Using both hands

With one hand on the mouse and another on the mother keyboard, **Add** a Note with the mouse, then place it at the correct pitch by playing a note on the mother keyboard.

Work area after End

Events after the End line in a sequence are ignored by the sequencer when playing the sequence. This fact can be used when building sequences in the Bar Editor. The area behind the End line can be used to hold melodic source motifs before Pasting. This is like having an extra Clip. It's also often quicker to drag unneeded events beyond the End line and **Delete** them later than to **Remove** them one by one while composing.

♦ Note: This technique causes the **Bars** indicator in the Sequence List (on the Sequencer Page) to appear larger than expected (because it calculates the length of the sequence according to time of the last event).

Jumping into Step record on the fly

Say you're recording live in the Editors and you make a mistake. Quickly press the SPACE BAR to **PAUSE** the clock. Then click **BACK** to erase the last recorded events.

Using RAM

This is an editing trick for other pages in Music-X. RAM disk can be used as a temporary holding area when copying Filters from one slot to another. Save the current filter to RAM as “temp”, change the current filter, and reload “temp” into the new slot. This can also be used when copying Keymaps or Protocols.

Song Assembly

This subchapter explains some of the various styles and techniques for assembling songs in Music-X. First of all, an overview of sequencing in general will do some good in orienting the reader to what's out there.

All the styles that have developed in music sequencing software can be organized into two very general categories which describe the far ends of the spectrum: Linear Sequencing and Segmented Sequencing.

Linear Sequencing is basically a descendant of analog recording procedures; the sequencer is treated as a virtual tape recorder. Linear Sequencing can be identified by these features:

There are a set number of parallel "tracks", each of which extends from the beginning of the song to the end, and does not repeat.

One instrument is recorded on each track; successive overdubs are put on separate tracks.

Everything is recorded from start to finish; each note is unique with no repeated sections.

If you have the playing ability, and enough computer memory to record all the parts from top to bottom, Linear Sequencing could be the quickest way to complete the piece, because it requires the least editing. It could possibly give you the most musical result as well, since each note is unique; you would be capturing a complete performance.

Segmented Sequencing is a descendant of drum machine programming and was originally intended to save memory by repeating sections whenever possible. The song is broken up into short segments or “patterns” which are played in various orders to make the finished product. Segmented Sequencing can be identified by these features:

Each segment of the song is recorded, complete with all the instruments, in its own “composite” sequence.

Only one sequence may be playing at a time.

Sequences are “linked” or “chained” in a series using some sort of “control track” or “song page” to arrange the form of the song after the fact.

The advantages to Segmented Sequencing are that memory is conserved and that the song form can easily be rearranged simply by changing the control track.

Illustration 15.1 shows examples of how a typical song might be assembled using the two methods. In the first example, each section of the song is recorded in its own sequence. A “control sequence” is then used to trigger the various song sections at the correct time, and in the correct order. In the second example, each instrument has been recorded on its own track. The rhythm section (piano, bass, and drums) plays continuously from the beginning to the end; the other instruments come in only when needed.

In practice, Linear and Segmented approaches have been implemented in myriad ways in different sequencers, often blending or overlapping into hybrid methods. Music-X is built on a more open structure that allows all of these and other methods to be combined and executed *simultaneously*. For instance, it’s quite possible to build a drum part by recording the kick drum, snare drum, and high-hat parts in segment style, then improvise a song-length drum fill sequence to play along with the basic tracks. Any sequence can be a “control sequence”, as well as be controlled itself. Music-X allows triggered sequences to overlap in time. Two or more instances of the same sequence may even be playing at the same time.

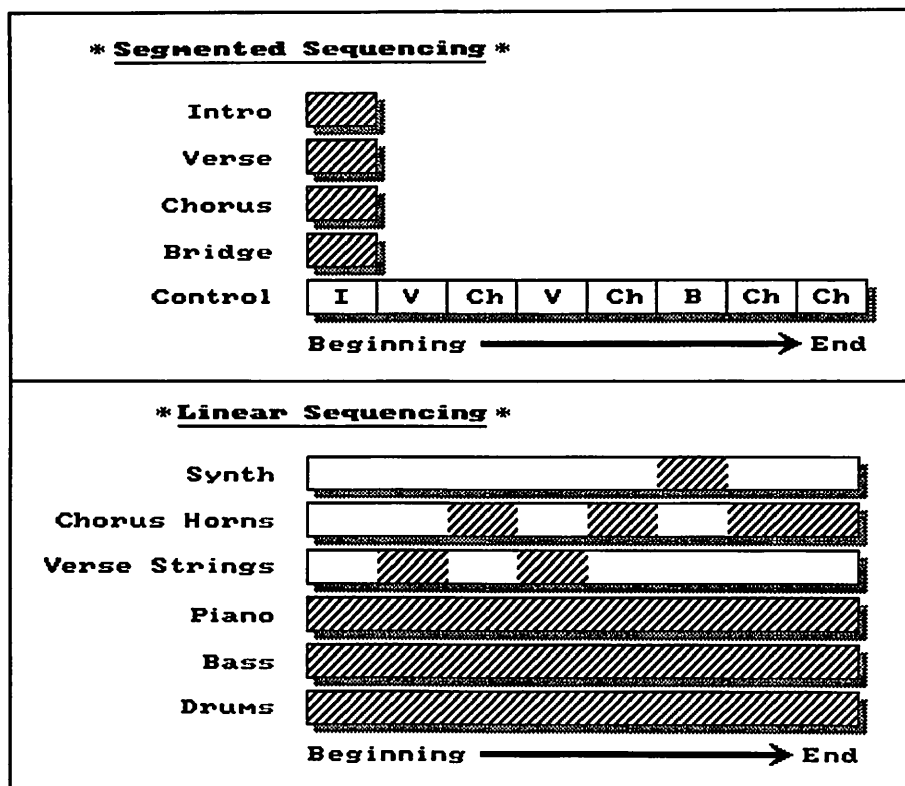


Illustration 15.1 Linear Vs Segmented Sequencing

As you can see, there are many ways to put together the structure of a song in Music-X. Just to get you started, a few are described in this sub-chapter.

Strict Linear sequencing

Strict linear-style sequencing is very simple.

- ☐ Record the first instrument and **STORE** it to sequence 1.
- ☐ Record the second instrument and **STORE** it to sequence 2.
- ☐ Etc.

Strict Segmented Sequencing

The basic process is to record each section of music on its own “composite” sequence, merging each instrument into the composite as it is added. Then turn off all the composite sequences, and use another sequence containing Play Sequence events as a control sequence to trigger the composite sequences.

- ☐ Record the first instrument on sequence 1.
- ☐ Record the second instrument on sequence 2.
- ☐ Use **Merge Sequence** in the Sequencer Page / Options Menu to merge sequence 2 to sequence 1.
- ☐ Record the third instrument, **STORE** it to sequence 2, and **Merge** it to sequence 1.
- ☐ Follow this process until sequence 1 contains all the instruments for the “Intro”, or whatever section you’re building. Then permanently mute sequence 1 by clicking its offset field to read **Off**.
- ☐ Move to a new sequence and build the other song sections in the same fashion (**RECORD, STORE, Merge**).

After all the source sequences are built, create a control sequence to trigger them one by one. This might be done in one of three ways.

- ☐ Manually assemble a series of PSEQ events in the Event Editor. The start and stop times would be easy to see this way.
- ☐ Or, assemble a sequence of Play Sequence events in the Bar Editor. The names are visible this way.

- ☐ Or, use a keymap attaching play sequence events to particular keys allowing you to trigger the source sequences from the mother keyboard, then record the control sequence in real time.

Relay Chaining

Another way to implement segmented sequencing is to chain sequences together by adding Play Sequence events at the end of each to trigger the next sequence. Each sequence “relays” control to the next sequence as it ends. A good example of this method can be found in the “Performances” directory on the Music-X Examples disk, in the performance titled “Toccata.Perf”.

- ☐ Record and **STORE** the first sequence. Then, with the editors, add a Play Sequence event just before the end of the sequence, triggering the second sequence.
- ☐ Record and **STORE** the second sequence. Turn it off. Then, add a Play Sequence event just before the end of the sequence, triggering the third sequence.
- ☐ Etc.

Improvise and Arrange

With cut and paste operations it's no longer necessary to be able to play your ideas perfectly in order to compose. It's possible to record a bunch of ideas “mistakes and all” and work with them in the editors to pull together a finished composition.

Here's a method of composing with Music-X that shows you how to capture a piece of music in its raw, inspired form and then carry it through the various editing stages towards a completed arrangement.

- ☐ Set up a metronome sequence and a tempo in the approximate feel of the idea in your head.

- ☐ Start recording and play along with the metronome, mistakes and all. It doesn't matter if you skip a beat or make your measures too long; we can fix that later. Just try to stay in the same pulse as the metronome.
- ☐ As you are recording, feel free to adjust the tempo.
- ☐ When you feel like stopping, click **STOP**. Then choose an empty sequence (click on it to highlight it), and click **STORE**.
- ☐ Save your work so far to disk. (Make a habit of saving after important steps are completed.)
- ☐ Use **Copy Sequence** to copy this sequence to another empty sequence as a working backup. If you don't have enough free memory, save your work again in a performance file called "improv.backup" or something. This is a copy of your initial improvisation. You may wish to listen back and compare when it's all done, or you may need to recall sections of it when editing.
- ☐ Now click **EDIT** and make sure you are in the Bar Editor.
- ☐ Play back your improvised sequence while you listen and watch for good parts, bad parts, mistakes, usable sections, unusable sections, etc.
- ☐ When you find a section you don't need, delete it. First, turn on **Snap** (this makes it easier to keep the length of the deleted section even). Then use **Mark** to highlight the unneeded section, and choose **Delete** from the Options menu.
- ☐ To rearrange the order of sections in the score, first **Mark** a section to be moved, then choose **Cut** and **Paste** from the Options menu to reinsert it at a new location.
- ☐ If you started playing behind the beat in a section, turn off **Snap**, then **Mark** and **Delete** a small amount of time at the beginning of the section to shift all following music earlier so that it sits closer to the beat.

- ❑ If you notice that you started playing ahead of the beat in a section, use **Insert** to insert a short dummy note into the sequence, pushing all following notes later and onto the beat. Then **Remove** the dummy note.
- ❑ To make fine adjustments in the position of groups of notes without changing any others, use **Select** to highlight the notes to be moved, turn off **Snap**, and press the CursorLeft and CursorRight keys.
- ❑ Use **Move** to correct the pitch or timing of individual notes.

Nested Sequence Triggering

Since each sequence may simultaneously be a control sequence and be controlled itself, control sequences may be “nested”. Sequences may control sequences that control sequences. Illustration 15.2 illustrates this concept as a hierarchical structure. Short musical patterns are played by sub-control sequences which are in turn played by a master control sequence.

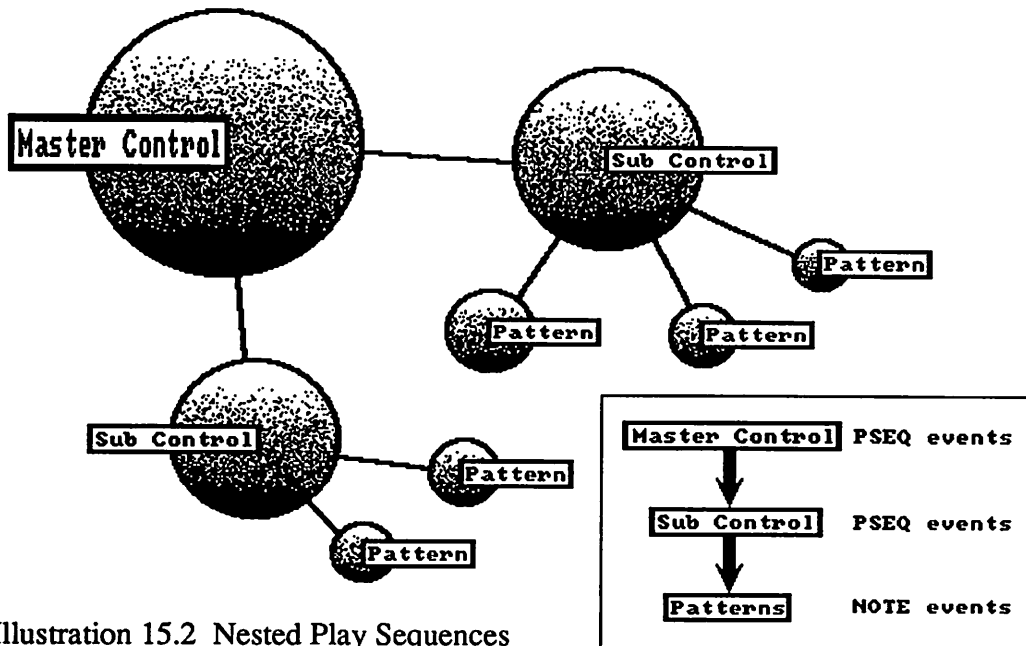


Illustration 15.2 Nested Play Sequences

For example, to build a hierarchical drum part:

- ☐ Create a bunch of single-measure drum patterns, and turn them all off.
- ☐ Using Play Sequence events, trigger the individual drum patterns in the correct orders from a sub-control sequence titled “Intro”. Turn off this sequence too.
- ☐ Build the other sub-control sequences containing Play Sequence events triggering the individual drum patterns. Title them “Verse”, “Chorus”, etc. and turn them off too.
- ☐ Now write a sequence called “Whole Song” that contains Play Sequence events triggering the sub-control sequences, creating the complete song. This is our “master control” sequence; a control sequence of control sequences.

Workbench Tool Types

This section describes how “.info” files, which are used to store Workbench icon information, can also be used to store additional startup parameters called “Tool Types”, which are used by Music-X.

Each Workbench icon is stored in a “.info” file (pronounced “dot-info” in programmer-speak), which contains the information needed by the Workbench to display the icon, and run the correct program when the icon is double-clicked. The file-names for “.info” files are formed by appending the characters “.info” to the name of the file or directory that they are associated with. For example, the icon for Music-X is stored in the file “Music-X.info”.

A “.info” file contains a picture, which is used to display the actual “icon” that you see, and other information about the file. There are three types of icons that are

important to us: A “DRAWER” icon is associated with a particular directory on the disk; a “TOOL” icon is attached to a runnable application, and a “PROJECT” icon is associated with a data file. In the case of “PROJECT” icons, there is a parameter called the “Default Tool” which tells the Workbench which application should be run when the icon is double-clicked. It is normally set to the name of the program that created the data file, but can be changed by the user.

In addition, the “.info” file can contain pieces of supplementary information which can be read by any program that opens this file (as Music-X opens performance files when it starts up). These parameters, which are like memos or instructions to the “Tool” (i.e., the program), are called “Tool Types”. You can add, delete, or edit Tool Types from the Workbench, and this will affect the way Music-X runs.

Tool Types are often used to supply settings and defaults for a particular program. In the case of Music-X, Tool Types are used to control decisions that are made when the program first starts up, such as how much memory to use, what certain defaults should be, etc. These are things that, due to the nature of the program, cannot be adjusted once the program is actually running.

When you start Music-X by double clicking on the “Music-X” icon, Music-X reads in the Tool Types for its own icon. When you start Music-X by double clicking on a performance file icon, Music-X reads the Tool Types for its own icon and the icon of the performance file. If both the Music-X program icon and the performance file icon contain contradictory Tool Type information, the performance file’s Tool Types will take precedence.

All Tool Types consist of a Tool Type name, in capitals, followed by an equal sign, followed by a string of text. Here are some examples of Music-X Tool Types:

```
QUIET=ON  
LIBDIR=df0:Libraries  
BUFFER=10000
```


To Edit the Tool Types of the Music-X icon:

- ☐ From the Workbench, click once on the Music-X icon. This will cause the icon to appear highlighted, indicating that it is “selected”. Do not double click on the icon, as you want to select it, not open it.
- ☐ Then in the Workbench Menu labeled “Workbench” choose the **Info** menu item.
- ☐ A window will appear on the Workbench screen, labeled “Info release 1.3”.
- ☐ Look for the **TOOL TYPES** box. If there are no Tool Types, then this window will appear empty, otherwise the first Tool Type will be shown.
- ☐ Use the up and down arrows within the **TOOL TYPES** box to view the list of Tool Types and select the one you wish to edit.
- ☐ Click with the mouse on the text of the Tool Type to activate the cursor, and edit the text as you would a normal intuition string gadget.
- ☐ When you are finished editing, click **SAVE** (in the lower left hand corner of the Info window) to save the new Tool Type parameters. Click **QUIT** or hit the close box on the window if you don’t wish to save the changes.

You can also add a new Tool Type, by clicking the **ADD** button in the Info window, which creates a new, empty Tool Type string for you to insert text into. You can delete the currently shown Tool Type by clicking on **DEL**.

You can also edit the Tool Types of a Music-X performance file, which is the same as the procedure for editing the Tool Types in the Music-X program icon, except that you would select the icon for the performance file instead of the Music-X icon.

These are Tool Types recognized by Music-X:

BUFFER

This Tool Type will set an upper limit on the amount of buffer memory that Music-X will use. Normally, Music-X grabs all the memory it can, but if you want to run other programs with Music-X, you either have to run them first, or use this option to limit how much memory Music-X will grab.

You can specify the amount of memory either in bytes (as in "BUFFER=50000"), or in kilobytes (as in "BUFFER=50K"). Music-X needs at least 30,000 bytes of buffer space to function properly, and will ignore a buffer limit lower than that, defaulting to its normal "grab all" mode. If you specify a number larger than the amount of available memory, Music-X will only take as much as it can.

PERFTOOL

This Tool Type is used to specify the Default Tool for any performances that will be created (the Default Tool tells Workbench which program to run when this icon is double clicked). When a performance file is saved, Music-X will create a ".info" file for that performance file, unless one already exists. The Default Tool in the ".info" file will be set to Music-X's PERFTOOL string. If PERFTOOL is not specified, it defaults to "Music-X:Music-X". Example:

PERFTOOL=df0:Music-X

QUIET

Normally, when Music-X starts up, it plays a little melody on all 16 MIDI channels to let you know that your equipment is turned on and functioning properly. However, in some situations this could be inconvenient, perhaps, shall we say, even embarrassing? For this reason the QUIET Tool Type has been provided to disable this feature. QUIET=ON disables the melody, QUIET=OFF (which is the default) enables it. Example:

QUIET=ON

EXPUNGE

This causes a “System Memory Panic” before booting Music-X. What that does is to flush out all unused devices, libraries, fonts, etc. that are taking up RAM, thus freeing up the maximum amount of memory. Since this is not normally a very “polite” thing to do to other programs, it has been left as an advanced user option which defaults to OFF. Examples:

EXPUNGE=ON
EXPUNGE=OFF

PERFDIR

This Tool Type sets the default drawer name in the “Load Performance” and “Save Performance” file requesters called from the Sequencer Page. Example:

PERFDIR=Music-X:

PROTODIR

This Tool Type sets the default drawer name in the “Set Protocol” File requester called from the Librarian Page, and the “Load Protocol” and “Save Protocol” file requesters called from the Protocol Editor Page. Example:

PROTODIR=Music-X:Protocols

LIBDIR

This Tool Type sets the default drawer name in the “Load Library” and “Save Library” file requesters called from the Librarian Page. Example:

LIBDIR=Music-X Examples:Libraries

FILTDIR

This Tool Type sets the default drawer name in the “Load Filter” and “Save Filter” file requesters called from the Filters Page. Example:

FILTDIR=Music-X Examples:Filters

MODLIST

This Tool Type tells Music-X where to find the file holding the list of installed modules. If not set, it defaults to “Music-X:MX.Modules”. Example:

MODDIR=dh0:MX.Modules

Installing Modules

Music-X has the ability to connect to independently produced modules, which are small programs designed to do a particular task for Music-X, but are not actually part of Music-X itself. For example, The “Quantize” Module, which is accessed from the “Modules” menu in the Event Editor and Bar Editor, is actually a completely separate program from Music-X. It would be possible to create a newer and better quantizer at any time, and replace the existing one, without affecting Music-X.

In order for Music-X to access a module, it must be “installed” with the “Install-Modules” program. This is found in the “Modules” directory on the Music-X Program Disk.

To run this program, double click on its icon, or the icon of any module.

Install Module Screen

On the top left of the modules screen is the “Modules:” control. This indicates the number of modules that are currently installed. You can edit this field by clicking on it, and then using either the plus (“+”) or minus (“-”) keys or the CursorUp and CursorDown keys to increase or decrease its value, or type in a number directly. To add a new module, you would increase this number by one, and then add the new module to the end of the list. Modules are deleted in two steps:

- ☐ If the module to be deleted is not at the end of the list, re-type the information for the module at the end of the list over the module you wish to delete.
- ☐ Decrease the number of modules by one. This deletes the last module in the list.

The lower portion of the screen shows a list of currently installed modules. You can scroll through this list using the up and down arrow gadgets. There are three columns of information: **Menu Text**, **Module File Name**, and **Type**.

Menu Text

This shows the text that will appear in a Music-X menu. The text may be up to 29 characters long. Any characters may be used, including Amiga “dead keys”, which are typed by using the ALT key in combination with other keys.

Module File Name

This shows the path name for each module so Music-X can find and load the new module when needed. The modules that come with Music-X are all in the “Modules” directory on the Music-X disk, referred to by a “logical assignment” called “MODULES:”. This logical assignment is created by the startup-sequence for the Music-X program disk with a statement that tells the Amiga’s operating system where to find the Modules directory:

assign MODULES: to df0:Modules

This creates a logical assignment with the name “MODULES:”. A logical assignment is the Amiga’s way of keeping track of where things are on a disk, even when that disk is not in the drive. Once you create a logical assignment to a disk, directory or file, AmigaDOS knows that the logical assignment is actually a shorthand name for that disk, directory, or file. For example, once the above statement was executed, AmigaDOS would know that “MODULES:Quantize” really means “Music-X:Modules/Quantize”.

◆ **Note:** If you take the disk out after the assignment has been made, the assignment sticks to the disk, rather than the drive. Thus, any reference to “MODULES:” will put a system requester saying, “Please insert volume Music-X: in any drive.” since that was the disk that was in df0: at the time that “MODULES:” was assigned.

You don't have to put all your modules in the same place, since you can specify a complete path specification for each module.

Type

This field indicates which of the five pages that have modules (Sequencer, Filter, Editors, Samples, Librarian) the module is attached to. If you wish to attach a module to more than one page, you must enter it twice. Note that the Bar Editor and Event Editor pages are considered a single Editor page for the purpose of installing modules.

Note that not all modules will work with all pages. Modules like "RunCLI" which don't directly access Music-X's capabilities can run from any page. "Quantize", "Scale Velocity" and "Scale Aftertouch" will only work from the editor pages, since they are designed to work on the edit buffer.

To edit the information for a particular module:

- ☐ Use the up and down arrow gadgets to scroll through the list of modules.
- ☐ Click on the module you wish to edit. A cursor should appear, and you can edit the string using the Amiga keyboard.
- ☐ The **Type** field can be edited by using the plus ("+") and minus ("-") keys, or the CursorUp and CursorDown keys to cycle through the various choices until you find the name of the page you want.

To save your new changes to and exit the Install-Modules program, use the menu item "Save and Exit". The Install-Modules program will save the list of installed modules in the file "Music-X:MX.Modules". The next time you boot Music-X, your modules will be shown in the menus.

To exit the Install-Modules program without saving any changes, use the menu item "Exit, No Save".

File Conversions

One of the recent additions to the MIDI specification is the Standard MIDI File (SMF) format. It suggests a standard way to save music data. The idea behind this standard is to allow people to edit the same piece of music using a number of different programs. A piece may start life in an artificial intelligence composer program, be imported into a sequencer program where it's improved, and finally be imported into a scoring program where it's printed out to be played live.

Unfortunately, SMF can't be everything to everybody. Some programs put more information into the music than SMF supports. Music-X is one of them. Music-X uses its own file format that allows the saving of things like Samples, Patch Libraries, Sync Settings, Keymaps, etc.

However, to provide compatibility with SMF, a program has been included on the Music-X Utilities disk that converts Music-X files to MIDI files, and MIDI files to Music-X files.

There is another music file format that is popular only in the Amiga world. This is the SMUS format. Many people have written and saved songs in this format using earlier Amiga programs. An additional program has been included on the Music-X Utilities disk to convert SMUS files to Music-X files.

Documentation on how to use the conversion programs can be found in text files on the Music-X Utilities disk.



Afterword

Thanks for taking the time to read the manual. If you've found any typos or glitches, please write to me at MicroIllusions so I can compile a fix sheet. Also if you've got any wild ideas or special wishes for Music-X version 2.0 let us know. We endeavor to stay contemporary.

Matt Nathan, Silverlake CA

Glossary

Absolute Time — Real time as opposed to musical time. Where musical time can change with tempo, absolute time moves at an even rate (modern physics ignored).

Aliasing — A sampling term meaning the audible interference between the sample rate frequency and the high frequencies in the sound source.

AmigaDOS — Amiga Disk Operating System. The one program in the Amiga that governs the operation of all other programs. The main program is called the operating system and all others are called *applications*.

Application — In a multi-tasking environment, each separate piece of *software* is considered an application. Music-X is one application.

ASCII — American Standard Code for Information Interchange. ASCII is a list of 128 characters including the letters of the alphabet in upper and lower case, the numbers 0 through 9, punctuation marks, and some characters that can't be typed like "bell". Each character is assigned a unique number between 0 and 127. This list of numeric codes, and the characters they represent, is used as a standard master alphabet when transmitting or storing text electronically.

Auto-locate — To relocate automatically, as when a tape transport controller fast forwards or rewinds to the exact spot needed, avoiding manual search.

BBS — Bulletin Board System. A shared computer data base accessed by telephone, generally used to "post" public messages, as on a bulletin board.

Binary — Base two numbering system. Using only the digits "0" and "1". Computer memory stores all information in binary format.

Bit — (BI)nary digi(T). The smallest unit of digital data, representing a number between 0 and 1.

Boot Up — From, "Raising oneself by one's bootstraps." To get started. Turning on the machine.

Buffer — A temporary area of memory used as a work space. In Music-X, this is known as the "Record Buffer", or as the "Edit Buffer".

- Bulk Dump** — Any type of MIDI System Exclusive message in which a MIDI device sends a copy of some portion of its memory.
- Button** — A screen *gadget* that activates some function when clicked, usually appearing as a word or text surrounded by a border.
- Byte** — Eight bits. A unit of data representing a number between 0 and 255 or one ASCII character. For example, the word “MIDI” takes four bytes, or 32 bits to store in ASCII format.
- Call** — Cause to appear or happen, as in calling a *requester* or *function*. This is actually a programming term as in “calling a subroutine”.
- CANCEL** — A button in *windows* and *requesters*, usually available whenever data is about to be changed. This offers a safe way to change your mind and back out of any process in the software.
- Cent** — 1/100 semitone. 1/1200 octave. A logarithmically derived constant, used to measure differences in pitch.
- Channel** — (see MIDI Channel)
- Chase** — To follow automatically, as when one tape recorder follows discontinuous time code coming from another (after the other has been rewound and played again), by relocating and continuing to play along.
- Choose** — An Amiga term. To pick an *item* from a *menu* by releasing the right mouse button while the item is *highlighted*.
- Click** — An Amiga term for *pointing* to some screen item, and then pressing and releasing the left mouse button.
- Clock** — 1) The finest *resolution* of *Relative* time differentiation. A division of a quarter note. In Music-X the resolution is 192 clocks per quarter note for Relative sequences. 2) As in “Master Clock”, an arbitrary timing reference.
- Close** — An Amiga term meaning to finish using a *window* or *requester* and remove it from the screen.
- Command** — 1) An Amiga term for any function that is activated immediately rather than opening a window or giving a safety prompt first. A command goes away as soon as it’s done (rather than staying active as with switches or modes). Tool words like **Unmark** are commands. 2) A “Music-X command” is one of the actions that can be initiated by a Keymap which tells Music-X

to do something that is usually initiated with a screen control (like a key equivalent).

Composite Sequence — In Music-X, a *sequence* containing information on more than one MIDI channel.

Control — Another name for *gadget*. Any of a group of screen devices manipulated with the mouse: sliders, buttons, etc.

Cursor — A small marker used when editing text that shows where the next character typed will appear. The cursor usually appears as a rectangle the size of one character, in a different color than your text. The cursor can be moved with the cursor keys, or by clicking anywhere in the text.

Daisy Chain — A necklace of daisies woven by wrapping the stem of one around the flower of the next. This term has been borrowed in computing to describe a method of transmitting data from one serial device to many. The devices are connected in a chain with the transmitter at one end. Data is passed down the chain by each device after receiving it.

Data — Information, especially in digital form.

Data Stream — See MIDI Data Stream.

Default — An automatic condition set by the software when no user choice has been made.

Device — For purposes of this manual, any unit capable of transmitting, receiving or storing digital information.

Directory — A way of organizing disk files in groups. Files are found “inside” directories. Directories may be *nested* inside other directories. (See also Drawer.)

Double Click — An Amiga term for two *clicks* in rapid succession. Used to run Music-X from the workbench. Also used to load and save files from the Music-X file requester.

Download — To receive data. The librarian can download patch data from synths. A MODEM user can download information from a BBS.

Drag — An Amiga term for *grabbing* some screen item, and then moving that item by moving the *mouse pointer* while still holding the LEFT mouse button.

Drawer — A Workbench object used to represent a *directory* on the disk.

Drawers are *opened by double clicking* on their *icons*.

Drive — Disk Drive. A unit capable of reading and writing information from a *disk*.

Drop Frame — A method of correcting the frame numbers of SMPTE Time Code to compensate for the slightly altered frame rate of American color television (29.27 FPS, 108 frames less than expected per hour).

Dump — See Bulk Dump.

Edit Buffer — Music-X's temporary memory area used by the various Pages as a "scratch pad". Also known as the Record Buffer. Music-X uses it when Recording, Editing, Extracting, Copying, or Merging a sequence.

Entry — A member in a list, as in "library entry"; one of the 16 separate bulk data dumps stored in a library.

Envelope — In geometry, the locus of intersections of a series of curves or surfaces. In electronic music, it refers to a pre-programmed shaping of volume or tonality over time.

EOX — (E)nd (o)f system e(X)clusive. A *byte* with the value F7 used to mark the end of a *system exclusive* message.

Event — An occurrence located in time. In Music-X, any of the low-level musical instructions that can be triggered by a *sequence*.

Field — 1) A control for displaying and editing text or numbers. 2) As in "video field", 1/2 video frame. (See also VBI.)

File — A clump of information on disk.

Filter — 1) In synthesis, a circuit or algorithm designed to lessen the intensities of selected frequencies in the sound. 2) The part of Music-X that can selectively lessen the quantity of recieved MIDI messages.

Flag — An indicator which the software checks to determine which of two alternative actions should be performed.

FPS — Frames per second.

Frame — A division of a second used in visual media, relating to the mechanics of movie cameras and video equipment.

- Front and Back Gadgets** — An Amiga term for the *gadgets* that appear in the upper right corner of *windows* or *screens*, allowing the window or screen to be displayed in front of or hidden in back of other windows or screens.
- Function** — In general, any particular thing that the software does.
- Gadget** — Any of the screen controls that can be manipulated with the mouse in some way. Things like *buttons*, drag bars, and *sliders*.
- Ghosting** — An Amiga Intuition technique used to indicate that a particular *gadget* or *menu item* is disabled and cannot be used. The item is masked by a patterned “screen” of pixels.
- Grab** — An Amiga term for pointing to some movable screen item, and then pressing and holding the LEFT mouse button. You grab something before you drag it.
- Handshaking** — The process by which a sender and receiver agree on intention and assure data integrity before or during data transmission.
- Hardware** — In computers, any piece of computer equipment. The physical equipment needed to run software.
- Hexadecimal** — Also “Hex”. Base sixteen numbering system using the digits “0” through “9” and the letters “A” through “F”. Hex is handy for expressing binary numbers in a more memorable, and therefore usable format. (see also Binary)
- Highlight** — An Amiga term for changing a screen object to a contrasting color, indicating that it has been selected. To cause to become highlighted.
- Hot Point** — The single pixel in the mouse pointer that is actually used when pointing or clicking. The hot point of the Music-X mouse pointer is at the tip of the chevron shape.
- Icon** — In the workbench screen; a small graphic picture representing a program or file. By *double-clicking* on the icon, a program can be caused to load and run.
- Indicator** — A display that shows the status of one or more parameters being used by the software.
- Item** — See Menu Item.
- Kilobyte** — 1024 bytes. 1024 is 2 raised to the 10th power. Powers of 2 are convenient for computers.

Librarian — Software used to collect and distribute *bulk data* used by MIDI Devices.

Library — In Music-X, a bundle of 16 bulk data dumps collected by the Librarian.

Linear — In a straight line. A sequencing method where each instrument part extends from the beginning to the end of the song. (See also Segmented).

Load — The opposite of Save. To receive data, specifically from a disk drive.

Locate — In Music-X, the process by which the sequencer finds the point in the performance at which to start playing. This is done by searching quickly through the sequences to be played. This also updates the synthesizer controllers to values appropriate to the particular point in the music.

Memory — With computers, a device or medium which holds digital data. Usually refers to *RAM*.

Menu — A list of text items that “pulls down” when the *mouse pointer* is pointed at the menu’s title in the *Menu Bar* while the right mouse button is held.

Menu Bar — The strip across the top of a *screen* that shows the names of available *menus* when the right mouse button is pressed.

Menu item — An Amiga term for one of the actual lines of text in a *menu*. A menu item is chosen by releasing the right mouse button when the menu item is highlighted.

Message — When speaking of communication between serial devices, a message is a string of binary numbers sent electronically from one device to the other representing a code that both devices understand.

MIDI — Musical Instrument Digital Interface. 1) A standardized language for communication between musical devices. 2) The serial interface hardware used by digital computers to speak MIDI.

MIDI Channel — A number from 1 to 16 that is attached to a MIDI message. (Channels are usually referred to as ranging from 1 to 16 although the actual MIDI messages use the numbers 0 through 15.) When a synth is set to transmit on some particular channel, it attaches that number to every message it sends from its MIDI OUT port. When a synth is set to receive data on a

certain MIDI channel, it still sees the whole MIDI data stream containing messages with various channel numbers attached, but will respond only to events tagged with its own channel number. This is something like giving each player in an orchestra a copy of the composer's complete score, from which they have to "fish out" their own individual parts.

MIDI Data Stream — The total stream of MIDI messages and data, transmitted serially in the form of bits over a MIDI cable. Instruments connected to the stream can choose to respond only to messages assigned to certain *MIDI channels*.

MIDI Device — Any piece of *hardware* capable of transmitting, receiving or storing MIDI information.

MIDI.device — The device driver software, used by Music-X, which communicates with MIDI instruments through the Amiga's serial port.

MIDI IN — This port is where a MIDI instrument or device receives the MIDI data stream.

MIDI Merger — A device that combines the MIDI OUT ports of multiple instruments into one data stream. This requires some intelligence since most MIDI messages are several bytes long and must be kept intact.

MIDI OUT — This port is where a MIDI instrument or device transmits data originating from itself. (Compare MIDI THRU.)

MIDI Sound Module — (see Sound Module)

MIDI Switcher — A device with two or more MIDI IN ports and one MIDI OUT. Designed to eliminate cable unplugging when changing input devices. This device allows you to choose which MIDI IN will be connected to the MIDI OUT. This is not the same as a *MIDI merger*, and may be implemented with rather simple electronics.

MIDI THRU — This port is used to re-transmit an exact copy of data received at the MIDI IN port. MIDI THRU can be used to *daisy chain* several MIDI devices together.

MIDI Thru Box — (see Thru Box)

MIDI Time Code — A type of MIDI message used to synchronize MIDI devices together, specifying Absolute time in hours, minutes, seconds and quarter-frames.

Mix Down — An audio recording term used to describe the process of playing back many pre-recorded tracks while simultaneously re-recording them onto a lesser number of tracks (usually two) on another machine. This is emulated in Music-X as Mix Down mode.

Mnemonic — A short word used to assist the memory. In Music-X, the abbreviations used in the Event Editor Page to name the various possible event types.

Module — In Music-X, a separate program that can be loaded into memory and accessed from Music-X. (See also Sound Module.)

Mother Keyboard — In a setup with multiple keyboards, the one used by the musician as the primary controller. Also used as the main input device when recording into a MIDI sequencer.

Mouse Pointer — An icon on the screen that moves as you roll the mouse around on its pad. The mouse pointer is used to *point* to *icons* or *gadgets* on the screen and do things like *click* them or *drag* them.

MTC — See MIDI Time Code.

Multi-timbral — Capable of producing the sounds of more than one instrument at once in response to MIDI messages received on multiple MIDI channels. Many sound modules can do this.

MX.Modules — The file that the Install-Modules program saves the list of modules and menu texts into. This file is read by Music-X on boot up to find out which modules to include in the various Modules menus.

Nested — A property of a set or series of similar things, such that each fits within the next largest one. A directory found inside a directory.

Non-Drop Frame — One of the standard SMPTE frame rates, corresponding to the NTSC standard of 29.97 FPS but counted as 30 FPS.

Note Off — A MIDI event that says "Release key number such-and-such."

Note On — A MIDI event that says "Press key number such-and-such."

NTSC — National Television Systems Committee (jocosely called "Never Twice Same Color"). Also, the frame rate of approximately 29.97 FPS.

- Nybble** — Four bits. Half a *byte*. The 16 possible combinations of four bits allow the forming of binary numbers 0000 through 1111. This probably had a lot to do with the original decision to have 16 MIDI channels. The channel number of a MIDI message is sent in the second nybble of the first byte in the message. In *hexadecimal* notation, a nybble can be represented by a single character. A byte, two nybbles, can be represented by two characters.
- Offset** — An arbitrary number added to another number or group of numbers. Example: Transposition up a step can be accomplished by adding an offset of 2 to the MIDI key numbers of all the Note events in a melody.
- Open** — In the case of Windows, Drawers and Applications, to make visible and active.
- Page** — 1) A full-screen display including *windows*, *gadgets*, and *indicators*, organized to control one general area of the software's operation (as in "Sequencer Page"). 2) A library file as it exists in memory and is displayed on the screen.
- PAL/SECAM** — PAL stands for "Phase Alternating Line" (jocosely called "Peace At Last"). The PAL system was invented in Germany and is a variation of the NTSC system. SECAM is a combination of French words that translate to "Sequential with Memory". PAL/SECAM represents the set of television broadcasting standards most widely used in Europe. Of primary interest here is the frame rate of 25 FPS.
- Parameter** — 1) A numeric control within Music-X whose value governs one particular aspect of the software's operation. 2) One of the attributes of an *event*. 3) In synthesis, one of many numeric controls whose setting governs an aspect of the timbre or functionality of the instrument.
- Patch** — Loosely, a synthesizer sound or setting. More accurately, a collection of all the parameter settings needed to get that particular sound on that particular synth. A holdover from the days when most synthesizers were modular and used *patch cables*. (See also Program.)
- Patch Cable** — A cable used with modular synths. So called because they allowed you to "patch in" various circuits on the way from the sound source to the amplifier, creating your own signal path.

Patch Dump — A System Exclusive message containing the data needed to recreate a particular timbre on a particular synthesizer. (see also Program)

Pixel — Short for “Picture Element”. One of a limited number of “dots” on the screen. The smallest area that can be individually displayed. A screen having a larger number of pixels is said to be higher in *resolution*.

Point — An Amiga term for positioning the mouse pointer on the screen so that its *hot point* appears to touch, or cover, some screen object (*icon, gadget, etc.*). Pointing is usually followed by *clicking* or *dragging*.

Pointer — see Mouse Pointer

Pointing — See Point.

Polyphonic — Literally, “many sounds”. When synthesizers were first invented, it was a big accomplishment to play one note. Later synths made the bigger accomplishment of playing more than one note at the same time. Since this was special, they gave them the name “Polyphonic Synthesizers”. Now the cool thing is to be *multi-timbral*.

Pop-out Menu — A secondary *menu* that appears to the side of another menu, while one of the menu items is highlighted, offering more items.

Preroll — When synchronizing audio or video tape machines using a prerecorded time code, preroll is the amount of time the transmitting unit must play (“roll”) while the other units catch up, before the system can be considered synchronized.

Program — On a synth, a program is the collection of parameter values needed to recreate a particular timbre. The word “program” is often used interchangeably with the words “patch”, “voice”, or “sound”.

Program Number — The number that identifies a particular stored sound (program) in a synthesizer.

Prompt — A message from the software, requiring some sort of user acknowledgement before proceeding.

Protocol — A bridge between the problem of multiple SysEx formats and the desirability of a single librarian. A protocol contains the extra information needed by the librarian to adapt to a particular SysEx format.

- Punch** — The technique of recording over a small part of a larger section of previously recorded material, erasing whatever was there before. Used to correct mistakes or build a sequence by sections. “Punch-In” is when you begin rerecording and “Punch-Out” is when you stop.
- Quarter frame** — A time division used in MIDI Time Code equal to one quarter of a frame. Actual time length depends on the current frame rate.
- Quick list** — A list of all the possible values for one particular *parameter*, any of which can be *clicked* on to set the value of that parameter.
- RAM** — Random Access Memory. The temporary erasable memory in the Amiga. This is where Music-X and its current performance resides when in use.
- RE-OUT** — In Music-X, the process by which data from the performer is received at the MIDI IN port, processed through filters and keymaps, and sent via the MIDI OUT port to sound modules, in real time. This is different than MIDI THRU since the data is echoed to the MIDI OUT port after being processed.
- Receive** — The opposite of *send*. To accept data; specifically from a synth or external device.
- Relative Time** — In Music-X, a way of measuring time relative to standard musical conventions, in other words using beats and measures. Tempo controls the relationship of Relative time to *Absolute Time*.
- Relocate** — See *locate*.
- Requester** — One type of *window*. A message from the software, requesting a decision or some sort of input from the user.
- Resolution** — The maximum accuracy in which an analog signal or value can be represented digitally.
- ROM** — Read Only Memory. Non-erasable memory in some computers and synthesizers. This is what holds the patches on synths that cannot be reprogrammed.
- Sample** — A digitized sound. A digital recording. To record digitally.

Sample Dump — A MIDI data dump containing a digitized sound. The MIDI “sample dump standard” allows samples to be transferred between samplers of different makes. Not all samplers support the MIDI sample dump standard, but may support their own sample dump format.

Sampler — A keyboard, sound module, or device, capable of recording and playing back samples.

Save — The opposite of Load. Writing data, specifically to a disk.

Screen — An Amiga term for a display dedicated to one *application*, typically with a *menu bar* and *front and back* gadgets.

Segmented — A sequencing method where song sections are recorded separately, then combined as segments. (See also Linear.)

Send — The opposite of *receive*. To transmit data; specifically to a synth or external device.

Sequence — A list of *events* to be played in sequential order at specific times.

Sequencer — A program or device which creates and manages *sequences*.

Serial — In a series. One at a time. A serial port sends or receives one bit at a time. MIDI is serial, and sends 31,250 bits per second (approximately 1000 note-ons).

Slider — A graphic screen control which can be “slid” by dragging it with the mouse, analogous to “slide pots” seen on most audio mixers.

SMPTE — (Pronounced “sim-tee”) More properly, SMPTE/EBU. Stands for “Society of Motion Picture and Television Engineers / European Broadcasting Union”. Also, the time code format standardized by these institutions. (See also NTSC and PAL/SECAM)

SMPTE Reader — A device that decodes SMPTE Time Code and provides the data to other devices.

Software — In general, information or programs used in a computer, as distinguished from the physical components (*hardware*). Software is to hardware as story is to book.

Sound Module — A stand-alone device that produces sound in response to received MIDI messages. A synthesizer without a keyboard. Also called a “tone generator”.

- Store** — To write data to memory. The sequence editor “stores” data to sequence memory.
- Striping** — Recording SMPTE time code onto recording tape is sometimes called “striping” the tape, presumably because if you could see the time code on the tape, the regularity of the messages would make it appear stripe-like.
- Subclock** — See Clock.
- Support** — The ability of something to use or be used by something else, as in, “My word processor supports spell-checking.”
- Sync** — Synchronize. To cause two or more time-critical devices (drum machines, sequencers, video tape recorders, etc.) to operate together.
- Synth** — A synthesizer.
- Synthesizer** — A device that creates sound electronically, having no mechanical sound device.
- SysEx** — System Exclusive.
- System Exclusive** — Also “SysEx”. The “loophole” in the MIDI specification that allows individual instrument manufacturers to design their own language extensions. SysEx messages must start with F0 (hex) followed by a manufacturer identification code. The body of the message may be anything but is limited to bytes with values between 00 (hex) and 7F (hex). SysEx messages are terminated with F7 (hex), or any other “status byte” ranging from 80 (hex) to FF (hex).
- Target** — In Music-X, the data item that is about to be updated or replaced by a particular operation, as in “target sequence”.
- Thru Box** — A device with one MIDI IN and many MIDI OUT ports. Used to eliminate delays caused by long *daisy chains*. The output of the computer or mother keyboard is fed into the MIDI IN of the thru box. The MIDI OUT ports are then connected to the MIDI IN ports of all the synths in the system. That way, every synth gets the data at the same time.
- Tick** — See Clock.
- Time Stamp** — A number, assigned to an *event* representing the particular moment in time at which the event occurred.

Toggle — To reverse the state of a switch or button.

Track — In Music-X, one of 20 “virtual playback heads” which are responsible for realizing the *events* in a *sequence* at the proper time.

Trigger — To initiate a process which is already set up to run.

Upload — To send data; specifically to a synth or to another computer. The Librarian Page uploads patch data to synths.

VBI — Vertical Blanking Interval. The time between each video “field” when the electron beam is darkened (“blanked”) during its return to the top of the screen. According to the NTSC standard, television screens contain 525 scan lines. In order to avoid flicker, every other line is scanned from top to bottom, then the electron beam returns to the top to fill in the other half of the lines. The beam is darkened while it travels back to the top of the screen to start over. This is called the “vertical blanking interval”. Each pass from top to bottom of the screen is called a “field”. There are two fields in each complete *frame*. The VBI happens twice per frame (slightly less than 60 times per second). The black bar seen on a television screen when the picture is rolling (as when the vertical hold is incorrect) is the vertical blanking interval.

Virtual — Having all the attributes and virtues of something without actually being that thing. A software emulation of a previously physical way of doing things. As in “virtual slider” or “virtual tracks”.

Voice — 1) A polyphonic part, as in “six voice synth”. 2) A patch or program, as in “the DX7II can store 64 voices in its internal memory”.

Window — A section of the screen display used to isolate and edit one or more parameters.



Appendices

Keyboard Equivalents	438
System Messages.	442
Protocol Language Summary	452
Music-X File Format	456

Appendix A

Keyboard Equivalents

SYSTEM MESSAGE REQUESTERS

RETURN for rightmost button or OK

SEQUENCER PAGE

ESC (Escape key) = All notes off

"B" = (B)EGIN

"E" = (E)ND

"1" = CUE(1)

"2" = CUE(2)

"3" = CUE(3)

"4" = CUE(4)

"=" = SET

SPACE BAR = PAUSE

"[" = Rewind

"P" = (P)LAY

"]" = Fast Forward

"R" = (R)ECORD

"S" = (S)TOP

SHIFT-"S" = (S)TORE

SHIFT-"E" = (E)DIT

SHIFT-"D" = (D)ELETE

SHIFT-"P" = (P)REVIEW

SHIFT-"R" = Clock: (R)elative Clock

SHIFT-"A" = Time: (A)bsolute Clock

RightAmiga-"." = Suspend Music-X

RightAmiga-"S" = (S)equencer Page

RightAmiga-"F" = (F)ilters Page

RightAmiga-"A" = (A)miga Samples Page

RightAmiga-"L" = (L)ibrarian Page

RightAmiga-"Q" = (Q)uit Music-X

RightAmiga-"P" = Load (P)erformance

RightAmiga-"V" = Sa(v)e Performance

RightAmiga-"O" = (O)utput Channelizer

Function key F1 = Set Punch-In

Function key F2 = Set Punch-Out

RightAmiga-"C" = (C)opy Sequence

RightAmiga-"M" = (M)erge Sequence

RightAmiga-"X" = E(x)tract Sequence

RightAmiga-"Z" = Toggle (Z)ero Origin

CursorUp = Move up through sequence list

CursorDown = Move down through list

SHIFT-CursorUp = Page up through list

SHIFT-CursorDown = Page down through list

RECORD REQUESTER

"G" = (G)O

"C" = (C)ANCEL

SEQUENCE SELECTOR

"O" = (O)K

"C" = (C)ANCEL

CursorUp = Move up through sequence list

CursorDown = Move down through list

SHIFT-CursorUp = Page up through list

SHIFT-CursorDown = Page down through list

SUSPEND WINDOW

RETURN (Return key) = RESUME

OUTPUT CHANNELS

"O" = (O)K
 "C" = (C)ANCEL
 "N" = (N)o, CANCEL

BAREEDITOR

ESC (Escape key) = All notes off

"/" = SetSig
 "A" = (A)dd
 "I" = (I)nsert
 "C" = Move (change)
 "V" = Remo(v)e
 SHIFT-"L" = (L)ock
 "E" = S(e)lect
 "M" = (M)ark
 "U" = (U)nmark
 HELP (Help key) = Restore last deleted event(s)
 "Z" = (Z)oom+
 SHIFT-"Z" = (Z)oom-
 SHIFT-"S" = (S)nap
 "Q" = (Q)uiet
 "L" = Scro(l)l
 "G" = (G)rid
 SHIFT-"P" = (P)arams
 SHIFT-"R" = (R)epet
 SPACE BAR = PAUSE
 "P" = (P)LAY
 "R" = (R)ECORD
 "S" = (S)TOP
 "J" = STEP
 "[" = BACK

RightAmiga-"L" = (L)oad Sequence
 RightAmiga-"S" = (S)ave Sequence
 RightAmiga-"N" = (N)ew Sequence
 RightAmiga-"/" = Select Sequence

RightAmiga-"T" = S(t)ore Sequence
 RightAmiga-"R" = (R)ecall Sequence
 RightAmiga-"X" = E(x)it Editor
 RightAmiga-"B" = (B)ar Editor
 RightAmiga-"E" = (E)vent Editor
 RightAmiga-"K" = Cut
 RightAmiga-"C" = (C)opy
 RightAmiga-"D" = (D)etele
 RightAmiga-"P" = (P)aste
 RightAmiga-"A" = Select (A)ll

CursorUp = Move current event up in score
 CursorDown = Move current event down in score
 CursorLeft = Move current event earlier in score
 CursorRight = Move current event later in score
 SHIFT-CursorUp = Move note up 1 octave
 SHIFT-CursorDown = Move note down 1 octave

EVENT EDITOR

ESC (escape key) = All notes off

"/" = SetSig
 "D" = (D)up
 "V" = Remo(v)e
 HELP (Help key) = Restore last deleted event(s)
 "M" = Select (think (M)ark)
 "U" = (U)nmark
 "N" = U(n)do
 "C" = S(c)roll
 "G" = (G)rid
 SHIFT-"P" = (P)arams
 SHIFT-"R" = (R)epet
 "Q" = (Q)uiet
 SPACE BAR = PAUSE
 "P" = (P)LAY
 "R" = (R)ECORD
 "S" = (S)TOP
 "J" = STEP
 "[" = BACK

Right Amiga-"L" = (L)oad Sequence
 Right Amiga-"S" = (S)ave Sequence
 Right Amiga-"N" = (N)ew Sequence
 Right Amiga-"/" = Select Sequence
 Right Amiga-"T" = S(t)ore Sequence
 Right Amiga-"R" = (R)ecall Sequence
 Right Amiga-"X" = E(x)it Editor
 RightAmiga-"B" = (B)ar Editor
 RightAmiga-"E" = (E)vent Editor
 RightAmiga-"K" = Cut (Kut)
 RightAmiga-"C" = (C)opy
 RightAmiga-"D" = (D)elete
 RightAmiga-"P" = (P)aste
 RightAmiga-"A" = Select (A)ll

CursorUp = Move up through event list
 CursorDown = Move down through event list
 CursorLeft = Move left through current event
 CursorRight = Move right through current event
 SHIFT-CursorUp = Page up through list
 SHIFT-CursorDown = Page down through list

QUANTIZER

"Q" = (Q)UANTIZE
 "C" = (C)ANCEL

KEYMAP EDITOR

ESC (Escape key) = All notes off
 "U" = (U)NDO
 "C" = (C)ANCEL
 "O" = (O)K
 "H" = S(H)OW
 "R" = (R)ANGE
 "S" = (S)ET
 "1" = Keymap (1)
 "2" = Keymap (2)
 "3" = Keymap (3)
 "4" = Keymap (4)

RightAmiga-"L" = (L)oad
 RightAmiga-"S" = (S)ave
 RightAmiga-"X" = E(x)it

AMIGA SAMPLES PAGE

ESC (Escape key) = All notes off

RightAmiga-"I" = Load (I)FF Sample
 RightAmiga-"X" = Load Soni(x) Sample
 RightAmiga-"V" = Sa(v)e IFF Sample

CursorUp = Move up through sample list
 CursorDown = Move down through sample list

LIBRARIAN PAGE

ESC (Escape key) = All notes off

"S" = (S)END
 "R" = (R)ECEIVE
 "A" = (A)LL
 "V" = PRE(V)IEW
 "P" = (P)rogram:
 "E" = (E)dit Entry

RightAmiga-"N" = (N)ew
 RightAmiga-"O" = L(o)ad
 RightAmiga-"V" = Sa(v)e
 RightAmiga-"C" = (C)lose

PROTOCOL EDITOR

RightAmiga-"L" = (L)oad
 RightAmiga-"S" = (S)ave
 RightAmiga-"X" = E(x)it

Appendix B

System Messages

This is a collection of possible messages from Music-X with their explanations. All messages except those generated by the File Requester appear in a small requester with one or two lines of text and a single **OK** button. Additional messages may appear from time to time from the Amiga's operating system. Those would be documented by Commodore-Amiga.

System Error Messages

ERROR: Can't find Diskfont.library

The "diskfont.library" file must be in your LIBS: directory in order for Music-X to run, as Music-X requires several special fonts.

ERROR: Can't find font *fontname*

Music-X needs the font specified in the message (*italicized text changes*).

ERROR: Can't find MIDI.device

Music-X requires the MIDI.device file to be in the "DEVS:" directory in order to run. This is the MIDI device driver which talks to the Amiga serial hardware.

ERROR: Can't load requested Module

Music-X attempted to load a module, and the specified file either didn't exist, or wasn't where Music-X was told it should be, or, AmigaDos had a problem loading the file.

ERROR: Can't open CIAA resource

There are two Complex Interface Adapter (CIA) chips in the Amiga, CIAA and CIAB. Music-X uses the high-speed timer on CIAA as the internal clock. This error message indicates that another program is currently using the timer, and it is not available.

ERROR: Can't open musicx.library

There is not enough memory available to initialize the musicx.library.

ERROR: Can't Open Screen

There is not enough Chip Memory available to open the Music-X screen.

ERROR: Can't Open Window

Music-X is unable to open the Music-X window on the Music-X screen, probably due to insufficient Chip Memory.

ERROR: Can't Read File

There was a problem reading the file, for some reason.

ERROR: Couldn't open port

There was insufficient memory to create the message port used to communicate with "MIDI.device", the MIDI device driver which talks to the Amiga serial hardware.

ERROR: Couldn't open the file

This indicates an attempt to open a file which either didn't exist, or had some other problem being opened. This usually occurs as a result of incorrectly typing a name into the File Requester's **File:** text box.

ERROR: Data in edit buffer is not a sequence

You can't store data into a sequence that isn't a sequence. The librarian, Protocol editor, and Patch Editors all use the edit buffer as well as the Sequencer Page.

ERROR: Empty Work Buffer

This indicates that the operation requires some data in the Edit Buffer in order to work. For example, you can't store a sequence with no data in it.

ERROR: Error Reading File

An AmigaDOS error occurred (bad sector, etc.) and the file could not be read.

ERROR: Error Writing File

An AmigaDOS error occurred (disk full, bad sector, etc.) and the file could not be written.

ERROR: File is not an IFF 8SVX sample

This occurs in the Samples Page, if you attempt to load a sample file from the File menu, using the items "Load Sample", and "IFF", and the file is not an IFF sample file.

ERROR: File is old Music-X format

Earlier beta-test versions of Music-X used a different file format. This occurs if you attempt to load a file in that format.

ERROR: Incorrect File Type

Each type of file in Music-X has a file type. This error will occur when attempting to load the TitlePage or a Patch Editor, and the file is not of the right type.

ERROR: Mangled Music-X File

The file you attempted to read was indeed a Music-X file, but the data in the file didn't make sense. This could indicate a damaged file.

ERROR: No musicx.library

There is not enough memory available to create the musicx.library within Music-X.

ERROR: Non-existent entry.

Indicates that you attempted to perform some operation on a non-existent library entry, which requires a library entry with some data in it in order to work.

ERROR: Non-existent page

Indicates that you attempted to perform, on a non-open library page, an operation that requires an open library page in order to work.

ERROR: Not a Music-X file

This occurs if you attempt to load a file which doesn't have the proper Music-X ID code.

ERROR: Not a valid Patch Editor File

The Patch Editor file you are attempting to call up doesn't seem to be a Patch Editor file, or the file is damaged.

ERROR: Not a valid protocol file

The file that you attempted to load is not a protocol file.

ERROR: Not an IFF file

This occurs if the "TitlePage" file is corrupt, or if the Patch Editor file you are attempting to call is not really a Patch Editor file.

ERROR: Not Enough CHIP Memory

Not enough Chip Memory available to load the sample. This can occur when loading a performance if not enough Chip Memory is available for samples. The performance will still load, but without samples.

ERROR: Not enough memory

There is insufficient memory to continue with this operation. Memory, in this case, can mean either buffer memory (memory which Music-X has already allocated for itself, and is managing internally), or free memory (memory which Music-X tried to

allocate, and failed). In the case of loading samples, you may have enough RAM, but not enough contiguous RAM. It's better to load the samples you need before editing sequences to avoid fragmenting RAM.

ERROR: Page must be opened before transfer

Indicates that **SEND** or **RECEIVE** was hit, and there was no currently open library page selected to transfer the data to or from.

ERROR: Patch Size Mismatch

Indicates that the patch data doesn't match the requested patch editor. Either the patch editor is the wrong one, has a serious bug, or the patch data has been damaged.

ERROR: Time Code Reader not ready

This means that Music-X could not open the time code reader device. (If you don't have one, that would explain it.)

ERROR: Too many Modules active

This is an internal error and should never occur in normal situations.

ERROR: Too many open pages

Indicates an attempt to open a library page when there are already 10 pages open. Note that loading or merging a performance with library subfiles saved in it will cause library pages to open.

System Warning Messages

Music-X Library still in use. Quit Function Aborted

Some other program has opened the musicx.library, and Music-X can't quit until that program has finished using it (because that would cause the other program to crash).

Nothing to Delete

You can't delete an empty sequence.

WARNING: File contains no Filter Data

This occurs when you attempt to load a filter file, or when you attempt to load a performance file as a filter file, and the file doesn't actually have any filter data chunks in it.

WARNING: File contains no Library Data

This occurs when you attempt to load a library file, or when you attempt to load a performance file as a library file, and the file doesn't actually have any library data chunks in it.

WARNING: File contains no Sample Data

This occurs when you attempt to load a sample file, or when you attempt to load a performance file as a sample file, and the file doesn't actually have any sample data chunks in it.

WARNING: File contains no Sequence Data

This occurs when you attempt to load a sequence file, or when you attempt to load a performance file as a sequence file, and it doesn't actually have any sequence data chunks in it.

System Notices

These messages indicate that a process is finished:

Extract Finished

The extract operation was completed successfully.

Merge Finished

The merge operation was completed successfully.

Librarian Errors

These errors are generated by the librarian during protocol transfers:

ERROR: Attempt to Change Constant Value

In a protocol string, numeric fields and certain protocol variables cannot be changed with the “=” expression operator.

ERROR: Can't send wildcard character

Wildcard characters can only be used in protocol strings received from the synthesizer.

ERROR: Device does not respond

This occurs if Music-X is waiting for a message from the synthesizer, and more than 5 seconds go by without getting one.

ERROR: Field length out of range

Numeric fields and wildcards are limited to a minimum field size of 1 and a maximum field size of 5.

ERROR: Internal Error

There was an internal problem with the librarian. (This may happen if you attempt to SEND and RECEIVE at the same time.)

ERROR: MIDI Communication Error

This indicates that there was an error in the serial hardware, and the Amiga “missed” a byte.

ERROR: MIDI.device not available

The Librarian was not able to access the “MIDI.device” file, either because some other program has locked it, or because the MIDI.device file couldn’t be found.

ERROR: No Matching Delimiter

A parenthesis or brace character was used in a protocol string, and no matching parenthesis or brace was found.

ERROR: No Matching Quote

A quoted string was used in a protocol string, and no matching quote was found.

ERROR: Nonexistent String

A protocol string referred to another (sub-string), which contained no data.

ERROR: Not enough free memory

Indicates that there wasn’t enough memory to complete the library transfer operation.

ERROR: Protocol Error

Indicates that there was a byte received from the synthesizer that the protocol didn’t expect.

ERROR: Syntax Error

There was a syntax error in one of the protocol strings.

ERROR: This protocol doesn't support ALL

Indicates that the protocol does not support the **ALL** button. (Protocols which require user interaction with the synthesizer's front panel to either send or receive can produce this message).

ERROR: Unknown Token Type

A character or symbol was encountered in the protocol string, such as "\$" or "!" and Music-X didn't have a clue as to what it meant.

File Requester Errors

These errors are generated and displayed by the file requester:

— Can't Find Directory —

The drawer specified in the **Drawer:** text box doesn't exist.

— Error Reading Directory —

There was a problem reading the disk.

— Not a Valid Directory —

The drawer specified in the **Drawer:** text box is actually a file, rather than a directory.

— Not Enough Memory —

There wasn't enough space in the memory buffer to build the directory list.

Appendix C

Protocol Language Summary

Field Types and Literals.

Literal Values

Hexadecimal number = numeric value

Singly quoted character(s) (") = literal ASCII value

Double quoted character string ("") = literal byte string

Expression Operators

+ = Add

- = Subtract (or Negative)

* = Multiply

/ = Divide

% = Modulus

! = Boolean NOT

& = Boolean AND

^ = Boolean Exclusive OR

| = Boolean OR

<< = Logical Bit Shift Left

>> = Logical Bit Shift Right

() = Parenthesized expression

=> = Assignment

Substring Codes

PR = Prefix Substring
PG = Program Substring
#1 = User String #1
#2 = User String #2

Variables (upper or lower case allowable)

J = Exclusive-OR checksum
K = Additive Checksum
M = Total Blocks Sent
N = MIDI Channel Number
O = Patch Offset
P = Program Number
Q = Number of Bytes to Send
R = Total Bytes Received
S = Total Bytes Sent
U = Total Blocks Received
V = Value Last Received

Data Type Codes

X = Send/Receive Normal Data
Y = Send/Receive Nybbleized Data

Misc Field Type Codes

** = Sentinel
@@ = Special Sentinel
FIN = Send Final Acknowledge after transfer complete
= Wildcard
W = Wait

Checksum Delimiters

{ = Begin Checksumming un-encoded data
 } = End Checksumming un-encoded data
 {{ = Begin Checksumming nybbleized data
 }} = End Checksumming nybbleized data

Message string syntax:

The following describes the syntax, or grammar of the message strings. Each description begins with the name of a syntax component, followed by a colon. Below each of these is the symbol or list of symbols which define the legal ways in which the syntax component can be formed. Groups of symbols in brackets (“[]”) are optional, ellipsis (“...”) indicate that more such symbols may follow, and italicized words represent other syntax components. The syntax components *misc-field-type-code*, *substring-code*, *data-type-code*, *variable*, *literal-value* and *expression* are defined in the previous section.

message string:

field [, *field*]...

field:

literal-value
substring-code
variable
misc-field-type-code
 (*expression*)
 (*expression.length*)
 (*variable.length*)
 (*misc-field-type-code.length*)
 (*data-type-code* [*length* [*offset*]] [:*total*])

length: *expression*

offset: *expression*

total: *expression*

Transfer Models:

Receive:

Init - Initiate - tell instrument to initiate data transfer.
Conf - Confirm - instrument indicates it's ready.
Ack - Acknowledge - tell instrument to send next block.
Data - Receive Data - data sent by instrument.

(Ack & Data repeat until end of transfer detected).

Send:

Init - Initiate - alert instrument to receive data transfer.
Conf - Confirm - instrument indicates it's ready.
Data - Send Data - data sent to instrument.
Ack - Acknowledge - instrument ready for next block.

(Data & Ack repeat until all data sent).

Appendix D

Music-X File Format

IFF, or Interchange File Format, has become a popular way of storing information on the Amiga and other computers. It has the advantages of flexibility, adaptability and extensibility.

The Music-X file format is an unregistered, private IFF form called MSCX. It is not intended to be a “general” format or any kind of a “standard” like the IFF ILBM Picture format — many of the attributes of the MSCX file would be inappropriate for other types of music programs.

The structure of a MSCX file is the same as any IFF file, beginning with a form header followed by a number of individual chunks. The format for IFF files is described in the IFF documents, which are available through Commodore-Amiga Technical Support.

[See Bibliography / EA IFF 85]

Note that the following discussion applies to all types of files used by Music-X except for two: Protocol Files and Patch Editor Files. These are separate and have different formats.

Note also that the format of MSCX is still evolving. For example, it is likely that the format of the Sequence chunk will be upgraded to allow things like text annotations. Fortunately, the IFF format allows improvements to be made while still maintaining upward compatibility.

Chunk Types

Chunk Name	Explanation	File Type Used In
AUTH	author name	Performance
NAME	song name	Performance
TSIG	time signature	Performance
TMPO	tempo	Performance
CUES	cue points	Performance
SEQU	a sequence	Performance, Sequence
FORM LIBR	a library	Performance, Library
FORM 8SVX	a sample	Performance, Sample
KMAP	a keymap	Performance, Keymap
OUTC	output channels	Performance
FILT	a filter	Performance, Filter
SYNC	sync settings	Performance
TCDD	time code settings	Performance
AUDI	audio filter setting	Performance

Note that the library and sample chunks are listed as **FORM LIBR** and **FORM 8SVX**. This is because these are stored as embedded forms, which are simply IFF forms which are stored as chunks inside another IFF form. This is explicitly allowed in the IFF specification.

Note also, that the chunks can come in any order, although it is preferred that the TSIG chunk come before any sequence chunks. This makes life easier for conversion programs.

Format of Chunks

AUTH (author name)

This is a simple ASCII string. The length of the string is the length of the chunk. Music-X only uses the first 30 characters, and will never save an author name longer than that.

NAME (song name)

Same format as AUTH; a simple ASCII string

TSIG (time signature)

2 bytes of beats per bar (bpb)

2 bytes of beat size (bsz)

0 = whole note

1 = half note

2 = quarter note

3 = eighth note

4 = sixteenth note

For you C language buffs, the formula for bar size is:

```
#define QUARTER_NOTE_TICKS 192
```

```
#define WHOLE_NOTE_TICKS (QUARTER_NOTE_TICKS*4)
```

```
barsize = bpb * (WHOLE_NOTE_TICKS >> bsz);
```

TMPO (tempo)

4 bytes of tempo, in quarter notes per minute

CUES (cue points)

6 cue points, each containing:

2 bytes of measure number of cue (zero origin)

2 bytes of clock count within the measure

AUDI (audio filter setting)

2 bytes:

0 = filter off. Otherwise filter on

KMAP (keymap)

2 bytes of keymap number (0-3)

564 bytes keymap tables

Keymap table format: 3 arrays of 128 bytes each. The first array holds the MIDI command byte for each key, then second holds the first MIDI data byte for each key, the third holds the second MIDI data byte for each key. Since there are 128 keys maximum in MIDI, and up to 3 bytes per MIDI command, $3 * 128 = 564$ bytes.

OUTC (output channels)

16 bytes, 1 per midi channel. Each byte contains the output channel (0-15) for that corresponding source channel.

8SVX (sample)

This is stored as a standard IFF 8SVX sample. Information about this format can be found in the IFF Documents available from Commodore.

[See Bibliography / EA IFF 85]

FILT (filter)

- 4 bytes of filter number (0-15)
- 1 byte of channel mode
 - 0 = No Thru
 - 1 = Re-Out
 - 2 = Internal
- 1 byte of keymap number (1-4, or 0 for no map)
- 1 byte of control map number (not implemented)
- 1 byte of Channel After Touch fraction (0-100)
- 1 byte of Polyphonic After Touch fraction (0-100)
- 1 byte of Pitch Bend Fraction (0-100)
- 8 bytes of input channel table

This table is used to re-map the channel number of incoming MIDI messages. The high nybble of the incoming status byte of the MIDI message, which indicates what type of message it is, is used as an index into the table. The entry in the table is a single byte which replaces the incoming status byte. This byte must be of the same event type, however the channel number may be different.

The last entry in the table is not used.

SEQU (sequence)

- 4 bytes of sequence number (0-249)
- 30 bytes of sequence name (only 27 bytes used in Music-X)
- 1 byte of start mode
 - 0 = sequence off
 - 1 = self-starting sequence
- 1 byte of clock mode
 - 0 = relative time
 - 1 = absolute time

- 1 byte of output bank
 - 0 = MIDI port
 - 1 = internal amiga voices
- 1 byte reserved
- 2 bytes of starting measure (zero origin, signed word)
- event list, 10 bytes per event
(See Event List format, below.)

SYNC (sync settings)

- 1 byte of relative sync type
 - 0 = Internal Clock
 - 1 = MIDI clocks
 - 2 = Derive Rel Clock from Abs Clock
- 1 byte of absolute sync type
 - 0 = Video Clock
 - 1 = MIDI Time Code
 - 2 = SMPTE Time Code
 - 3 = Derive Abs Clock from Rel CLock
- 1 byte of output sync enable
 - 0 = No MIDI Sync Out
 - 1 = MIDI Sync Out
- 1 byte of system start mode
 - 0 = Send MIDI Start/Stop
 - 1 = Receive MIDI Start/Stop
 - 2 = Ignore MIDI Start/Stop

TCDD (time code settings)

- 2 bytes of SMPTE rate
 - 0 = Non-Drop Frame
 - 1 = 25 FPS
 - 2 = 24 FPS
 - 3 = Drop Frame
- 4 bytes of timecode offset, in quarter-frames

LIBR (library)

LIBH chunk - library header

32 bytes of library name

32 bytes of protocol name

2 bytes of name offset (0-9999)

2 bytes of name length (0-31)

4 bytes reserved

protocol strings - each string preceded by length of string, in bytes

LENT chunk - library entry

30 bytes of entry name

1 byte reserved

1 byte channel number (0-15)

1 byte program number (0-127)

1 byte reserved

(chunklength - 34) bytes of System exclusive data

Event List Format:

An event list is stored as an array of "events", which have the following format:

3 bytes start time of this event

1 byte Command byte

1 byte 1st data byte

1 byte 2nd data byte

3 bytes stop time of this event

1 byte 3rd data byte

total = 10 bytes

Start time and **Stop time** have the following format for relative sequences:

- 12 bits measure number of this event (0-4095)
- 12 bits clock ticks in that measure (0-4095)

For absolute sequences, the format is like this:

- 24 bits quarter frame number of this event

Here are the meanings of the various fields for each event type:

Type Event	Command	Data 1	Data 2	Data 3
End	\$00	—	—	—
Stop	\$01	—	—	—
Repeat	\$04	repeats	—	—
Time Sig	\$05	numerator	denominator	—
Splice	\$06	—	—	—
Change Map	\$10	channel #	map #	—
SYSX Data Event		special		
Play Seq	\$70	sequence	transpose	cut
Set Tempo	\$72	tempo0:7	tempo8:14	—
Mute Track	\$74	track #	—	—
Solo Track	\$75	track #	—	—
Mute Sequence		sequence #	—	—
Note Event	\$8n	pitch	attack Vel.	release Vel.
Poly A.T.	\$An	note	pressure	—
Control Change	\$Bn	control #	value	—
Program Change	\$Cn	program #	—	—
Channel A.T.	\$Dn	pressure	—	—
Pitch Bend	\$En	bend0:7	bend8:14	—

In the above example **n** represents the channel number for this event.
Both Tempo and Pitch bend data values are stored as 14-bit numbers, the lower 7 bits in data1 and the upper 7 bits in data2.

SYSX Data Events have a different format than other event types:

3 bytes starting time of event

4 bits command = \$3

1 bit halt bit

0 = more data events in this message

1 = this is the last data event in this message

3 bits how many bytes of the message are in this event (max 6)

6 bytes the actual data bytes of the message

total = 10 bytes

Also, note that the command byte of a **SYSX** data event is not the same as the **MIDI** command byte for a System Exclusive message.

Index

- #1, SUB-STRING Panel 371
- #2, SUB-STRING Panel 371
- (dir) 34
- .info 39
- A. Velocity 253
- About Box 47, 54
- About Music-X 57
- Absolute Time Sequence 80, 99, 102, 117, 162,
179, 184, 212, 217, 222, 223, 249
- accelerando 183
- Ack:, RECEIVE Panel 370
- SEND Panel 371
- Action List, Keymap Editor 308-314
- actions 294
- Add tool word 198-200
- Aftertouch Scaling Module 157-161
- ALL button, Librarian 356
- All button, Quantizer 151, 152
- All Notes Off 287
- Alpha keys 38
- Amiga Basics 389-395
- Amiga operating system 41
- Amiga samples 15
- Amiga Samples Page 27, 48, 319-333
- AmigaDOS 90, 91
- AmigaShell 91
- Arrangement 407-409
- Articulation 153, 401-402
- ASCII format 368
- ASCII strings, protocol language 377-378
- Assembling a song (see Song Assembly)
- Attack and Release Velocity 169, 253-254
- attack velocity 152, 155, 159
 - display 144-145
 - scaling 155, 226
- Audio Filter 55, 324
- auditioning a sequence 216
- author 47, 54
- Author Name 47, 54
- Auto Punch-In 101
- Auto Scroll 211
- Auto Step 134, 135, 225, 229-232
- auto-repeat 220, 268
- auto-scroll 267
- Available ChipMem: 329
- AVL: 169
- BACK button 227, 234
- backup, disk 3, 4, 5
- Backwards.Kmap 316
- Backwards2.Kmap 316
- Bank, Output 115
- Bar Editor 25, 48, 119, 126-235, 239
- Bar field, Track Window 117
- Bars field, Sequence List 113, 117
- beat numbers 162
- beats in a measure 111, 185
- Beats: slider 259
- BEGIN button 24, 92
- Bend Value (BND:) 178, 256
- blank sequence 130
- block, defining 138
- blue notes 177
- BND: slider 178
- boot 9, 23-24, 50
- buffer size 9, 110, 413
- Buffer, Edit 61, 65, 118, 119, 128-132
- Buffer, Record 28, 122, 128
- BUFFER= tool type 413
- bulk data, receiving 353, 354
- Bulk Dump 348, 349, 353, 359, 368
 - definition 423
- Button 423

- Buttons, using 391
- Byte 423
- Call 423
- CANCEL 423
- CANCEL button, Keymap Editor 299
- Cancel:, SUB-STRINGS panel 371
- CAT (Channel Aftertouch) event 254, 63
- Cent 423
- Ch (Channel Number), Event List 249
- Ch (Channel Number), Sample List 325
- Change Keymap (KMAP) event 186-187
- Change Keymap, from Keymap 314
- Channel A.T., Display 145
- Channel Aftertouch 63, 157, 170-171
 - Aftertouch Scaling 160
 - events 145, 157, 159, 171
- Channel events 251-252, 63, 65, 166, 167-178, 250
- channel numbers 113, 251
 - remap 282
- Channel Square 141, 142, 168, 171, 177, 203, 205, 232-233
- Channel: buttons
 - Filters Page 280-281
 - Librarian Page 357
- Channel: quick list 249, 251, 269
- Channelizer 55, 60
- channels (MIDI) 115
- Channels, send and receive 28
- Channels, Sequence List 113
- Char Map: 368
- Chase 423
- Chip Memory 67
- CHN: slider 187
- Choose 423
- Chords 231
- CLI 39, 90, 91
- Click 423
- Clip 136-143, 209
 - cutting to 243-244
 - copying to 244
 - Event Editor 243-246
 - paste from 245
 - show clip 246
- Clock
 - advance and rewind 269
 - advancing 135
 - definition 423
- clock number 162
- Clock Window 107-108
- Clock:, changing 107-108
- clocks, grid size 212
- Close (library) 342
- Close 423
- Command 423
- Command Line Interpreter (CLI) 90
- composite sequence 102, 113, 202, 424
- Confirm:, Protocol Editor 370
- Control 424
- Control Change (CTL) events 63, 145, 173-175
 - Display 146
 - from Keymap 310-312
- control events 113, 114
- control sequence 114, 187
- Control: slider 255
- Controller Number (CTL:) 174, 255
- Controller Value (VAL:) 174, 256
- controllers 173
- conversion, SMPTE to MIDI 75
- conversions, music file format 52, 419
- copies 3, 4
- Copy 139-140, 244
- Copy Sequence 61-62
- copying, disk 3, 4, 5
- Count-In Bars 102, 104-105, 116, 226
- Create Protocol 343

- Creating Envelopes, Amiga Samples 332-333
- crescendos 154, 156, 159
- CST, Sequence List 113-114
- CTL (Control Change) 63, 255-256
- CUE buttons 92-93
- current event 127, 164, 239
- current filename 37
- current sequence 28, 112, 122, 132
 - editing 119
 - erase 121
 - number 131, 221-222
 - changing 222
- cursor 114, 424
- cursor keys, moving events 165
- Cut (to clip) 136-139, 243-244
- Cut Notes, Keymap Editor 310
- Cut priority, Play Sequence event 190
 - cut= - - 261
 - cut=All 261
 - cut=Seq 261
- Cut Sequence, Keymap Editor 310
- D= indicator 141, 142, 165, 170, 189, 191, 193, 194, 196, 206, 223
- Daisy Chain 18-19, 424
- Data 424
- data dumps 363
- Data Echo: switch 288-291
- Data stream 424
- data, block 137
- Data, field 250
- Data:, Protocol Editor 370
- Data: slider 257
- decelerando 183
- decrecendos 154, 156, 159
- Default 424
- default target sequence 131
- Default.Perf 50, 53
- DELETE button 121
- Delete block 140, 141, 244, 245
- Demol.Perf 24
- demos 24
- Denominator (DEN:) slider 185
- Denominator 260
- Device 424
- DF0: (drive button) 35
- directory 424
- directory names 34
- DISCARD 25, 120, 133
- disk 35
 - backup copies 3-6
 - data 8
 - protection 3
 - rename 6
- Display Menu 144-146, 246
- dot (musical) 214-215
- Double Click 424
- Download 424
- Drag 424
- dragging 164
- Drawer, definition 425
- Drawer: (text box) 37, 38
- drive buttons 35-36
- drives 35-36, 425
- Drop Frame 81, 249, 425
- drum breakdown 192
- drum machine 20, 71, 72, 217, 218
- drum pad 21
- Drum Patterns 399
- Dump 425
- Dup tool word 241, 264
- Duplicate disk 3-6
- Dur: 235
- dur= 252
- duration (events) 246
- Duration 147, 151, 165, 170, 183, 189, 212, 213, 227, 229, 235, 246, 252-253
 - definition 213
 - display 246
 - editing 253

- Duration Only button, Quantizer 149
- dynamics 154
- E+ and E- 331
- echo 288
- Echo, Sample Envelopes 288, 289, 333
- edit
 - buffer 118, 119, 128-132
 - current sequence 119
 - multiple events 127
 - sequence 161
 - single event 127
- Edit Buffer 61, 65
 - definition 425
- EDIT button 25, 119-120, 128
- Edit Entry 343-344
- Edit It button 120, 128
- Edit Keymaps 280, 293
- Edit Menu, Editors 134-135, 242
- Edit Menu, Librarian Page 342-344
- Edit Panel, Amiga Samples Page 327-331
- Editing Tricks 399-402
- editor 126
- editor pages 134
- editors 25
- Effect %: slider, Quantizer 151
- elapsed time 107
- ENABLE switches 284
- Enable Sync 217-218
- End 178-179
- END button 92
- End event 130, 179, 189, 221, 257, 402
- End Line 164, 179, 189, 221
- End of System Exclusive byte 257
- End Times, Display 246
- Entire Sequence, Aftertouch Scaling 160
- Entire Sequence, Velocity Scaling 156
- Entry 425
- Envelope 425
- Envelope Editor, Amiga Samples 329-333
- envelope generators 153
- EOX byte 257, 425
- Equipment 3
- Erase All Sequences 66-67
- erase, current sequence 121
- ERR: indicator 110
- Error Spectrum 147-148
- event
 - adding and inserting 144
 - current 127, 164, 239
 - definition 127, 425
 - deselect 265, 266
 - duration 246, 252-253
 - end time 246
 - parameters 127, 250
 - selected 205
 - start time 248, 251
 - stop time 246, 252
 - type 144, 249
 - type descriptions 166-196, 253-262
 - velocity 154
- Event classes
 - Channel Events 63, 65
 - Music-X events 63, 64, 113
 - System Exclusive Events 63, 64
- Event Editor 25, 48, 119, 127, 134, 237-271
 - adding and removing events 241
 - Overview 238-239
 - using the keyboard 240
 - using the mouse 240
- Event field 248
- Event List 239, 240, 247-250
- Event Type, Remap panel 282
- Event Type: quick list 240, 249, 250-262, 269
- Ex1 115
- Examples disk 3, 24
- Exit
 - Bar Editor 133-134
 - Keymap Editor 298
 - Protocol Editor 367

- EXIT button 25
- Expressions, protocol language 378-379
- EXPUNGE= tool type 414
- External (Output Bank) 115
- Extract Sequence 63-66
- Fast Forward button 95-96
- Feedback 136-137, 210, 242-243, 266
- Field 425
- File 425
 - file conversions 52, 419
 - file extensions 39
- File Menu 50-57, 129-134, 241, 278-280, 296-298, 321-324, 340-342, 366-367
 - definition 31
- File Requester 29, 31-42, 52, 53, 56, 57, 129, 130, 297
 - definition 31-42
- File: (text box) 37, 38
- filenames 37, 40-41
- FILTDIR= tool type 415
- filter 425
 - controls 284-285
 - current 275, 281
 - data path 276-277
 - target 279
- Filter Settings (subfile switch) 54
- Filters Page 26, 48, 273-291
- Final Level slider, Aftertouch Scaling 159
- Final Level slider, Velocity Scaling 154
- Flag 425
- floppy drives 36
- footnotes 12
- Format indicator 346
- FPS 425
- Frame 425
- frame rate 80
- Free Largest Octave 323-324
- free memory 4, 5, 9, 52
- Free Sample 323
- FROM column 58
- Front and Back Gadgets 426
- Function 426
- Gadget 426
- Generic Patch Editor 359-360
- Generic.Multiple protocol 354
- Generic.Single protocol 354
- Ghosting 426
- Grab 426
- Graphic Score Window 161-165
- Grid 147, 208, 212-216, 267
- Grid Requester 193, 194, 196, 210, 213-214, 230, 267
- Grid Size 147, 150, 165, 168, 178, 188, 191, 194, 205, 206, 212-216, 227, 230, 267
- Grid tool word 147, 212-216, 267
- Grid: indicator 235
- guard bands 77
- handshaking 348, 351, 372, 373
 - definition 426
- hard disk 36
- Hardware 426
- HELP key 265
- hex data 359
- hexadecimal 286-287, 360
 - definition 426
- High Note, Sample List 327
- Highlight 426
- HOLD button 122, 134
- Hot Point 426
- icon 5, 8, 24, 39, 53, 130
 - .info files 9
 - definition 426
 - filenames 34
 - performance files 130
 - protocol files 367
- IFF (samples) 322

- Ignore (MIDI Start & Stop) 79
- Improvisation and Arrangement 407-409
- Index (Thumb) 13
- Indicator 426
- info files (see icon)
- Initial Level slider, Aftertouch Scaling 158-159
- Initial Level slider, Velocity Scaling 154
- Initialize All (filters) 279
- Initialize, disk 8
- Initiate:, Protocol Editor 370
- Insert tool word 200-201
- Install Module screen 416-419
- Int (output bank) 115
- interface, MIDI 3, 16
- Internal (master clock) 70
- Internal Amiga Samples 115
- INTERNAL button 289-291
- Internal Clock 217
- internal memory 36
- Item 426
- key number 162, 169, 172, 294
- KEY: slider 189
- key= 253, 255
- keyboard equivalents 13, 438-440
- keyboard map 274, 275, 294
- Keyboard Map: switches 287-288
- keyboard notes 161
- keyboard, MIDI 3, 17
- keyboard, mother 17, 19-21
- keymap 274, 275, 294, 296-297
 - load 297
 - name 298
 - save 297
- Keymap Editor 26, 186, 293-316
 - Action List 308-314
 - Editing Steps 295
 - MIDI Recording 314-316
 - Miniature Keyboard 302-308
 - Mode Panel 298-301
 - Parameter Window 314
- Keymap event 260
- Keymap Number 187
- keymap ranges 294-301, 302
 - defining 302
 - delimiters 302
 - SET 301
- keymap settings, SHOW 300
- Keymaps (subfile switch) 55
- keys, miniature keyboard 302
- Kickstart 4, 5
- Kilobyte 426
- KMAP (Change Keymap event) 260
- Largest: (Chip Memory) 329
- length of piece 108
- length of sequence 113
- Length: slider 257
- LIBDIR= tool type 414
- Librarian 427
- Librarian Page 27, 49, 335-360
 - Page Displays 357-358
- Libraries (subfile switch) 54
- Library 357, 427
- library data, send & receive 346
- Linear 427
- Load 31, 427
 - Filter 279
 - Keymap 297
 - Library 340
 - Performance 24, 51-52
 - Protocol 366
 - Sample 322
 - Sequence, from Bar Editor 129
 - Sequence, from Sequencer Page 56-57
- Local Control On/Off 287
- Locate 427
- Lock tool word 144, 202-203, 233
- Lookout Utility 376
- Loop Mode On 101-102, 180

- makedir 39
- MAP buttons 301
- MAP: slider 186-187
- Mark tool word 138
- Marked Events, Aftertouch Scaling 160
- Marked Events, Velocity Scaling 155-156
- Master Clock 70-77
- Max Threshold: slider, Quantizer 151
- measure 161
- Measure Numbers 162
- Mem (Memory used), Sequence List 113
- Memory 427
 - chip 67
 - free 4, 5, 9, 52
 - RAM 3, 113
- Memory: indicator 110, 217
- menu 4, 427
 - using 389-390
 - titles 129
- menu bar 46, 129, 296, 427
- Menu item 427
- Menu Text, Install Module 417
- Merge Performance 50-51
- Merge Sequence 61-63
- merger, MIDI 21, 350
- Message 427
- message strings (protocols) 365, 372, 376
- metronome 28
- Metronome Track, setting up 396-397
- middle C 169
- MIDI 395-396, 427
 - bus 115
 - control codes 285-287
 - data stream 428
 - device 428
 - formats 372
 - keyboard 3, 17
 - sequencer 45
 - setups 15-21
- MIDI Channel Number 172, 249, 427
- MIDI Channels 58, 113, 115, 141, 142, 157, 168, 249, 280
- MIDI Clock messages 70, 71, 73
- MIDI Clocks 70-72, 217
- MIDI Continue messages 78, 217
- MIDI Delay 188, 400-401
- MIDI Doubling 400
- MIDI IN 16, 73, 428
- MIDI interface 3, 16
- MIDI key number 162, 169, 172
 - Polyphonic Aftertouch events 172, 255
 - Note events 253
- MIDI Key#: indicator 308
- MIDI merger 21, 350, 428
- MIDI Message panel 285-287
- MIDI Messages 250
- MIDI OUT 16, 73, 288, 289, 428
- MIDI Recording 314-316
- MIDI Start & Stop option 78-79
- MIDI Start and Stop messages 78, 217
- MIDI switcher 350, 428
- MIDI Sync Out 83-85
- MIDI THRU port 18, 428
- MIDI Time Code 74, 75, 428
- MIDI Volume effects 401
- MIDI.device 428
- Min Threshold: slider, Quantizer 150-151
- Miniature Keyboard, Keymap Editor 302-308
- Mix Down Mode 102-103, 429
- Mnemonic 429
- Mode Menu 47-49, 278, 321, 339
- Mode Messages 174-175
- MODEM 16, 364
- Modify Protocol 343
- MODLIST= tool type 415
- module 90, 429
- Module File Name, Install Module 417-418
- Modules Menu 90-91, 146, 247, 280, 346
- Modules, Installing 416-418
- Mono Mode 287

- mother keyboard 17, 19-21, 429
- mouse pointer 5, 429
- mouse, dragging 164
- Move tool word 201
- moving events with cursor keys 165
- MSEQ (Mute Sequence) 261
- MTRK (Mute Track) 262
- Multi-timbral 429
- multitasking 52
- Music-X Command, from Keymap 313-314
- Music-X events 63, 64, 113, 146, 167, 233, 249
- Music-X Events switch, Extract Sequence 65
- musical time 161
- Mute Sequence (MSEQ) 114, 118, 190-191
- Mute Target Sequence 103
- Mute Track (MTRK) 114, 193-194
- Mute Track, from Keymap 312
- MX.Modules 429
- Name Length: 368
- Name Offset: 368
- National Television Systems Committee 80
- Nested 429
- New (library) 340
- New (sequence) 130-131
- New Tempo parameter 183, 259
- NewCli 90-91
- NewShell 91
- newtempo= 259
- No Cut 310
- Non-Drop Frame 80, 249, 429
- non-MIDI devices 16
- Normal Program: 367
- note
 - display 144
 - middle C 169
 - numbers 189
- Note events 63, 144, 152, 155, 164, 167-170, 253-254
 - memory usage 217
 - transmutation 274, 280
- Note Off 150, 152, 199, 429
- Note On 149, 150, 152, 199, 429
- note values (Grid requester) 213
- NT: 169, 172
- NTSC 80, 429
- NUM: slider 185
- number bar, virtual sliders 271
- Numerator (NUM:) slider 185
- Numerator 259
- Numeric Variables, protocol language 379-382
- nybble 60, 430
- Oct (Octaves), Sample List 327
- octal 176
- octaves 163-165, 302
- Offset arrows, Sequence List 116
- Offsets 189, 209, 399-400, 430
- OK button, Keymap Editor 300
- Omni On/Off 286
- Open 430
- Options Menu 58-67, 135-143, 242-246, 324, 345
- Out (Output Bank) 115, 137, 243
- Output Channelizer 55, 58-60
- overdub 102
- Overlap.Kmap 315
- Page 430
- Page Displays 357-358
- page name 278
- PAL/SECAM 430
- Parameter Window, Keymap Editor 314
- parameters 127, 250, 430
- Params tool word 216-220, 268
- pass through 16
- Paste 136, 138, 139, 141-142, 245, 399
- PAT (Polyphonic Aftertouch) event 63, 145, 157, 171-173, 254-255
- Patch 430
- Patch cable 430
- patch data, transfer 346

- Patch Dump 431
- Patch Editor Name: 368
- Patch Editor Pages 27
- Patch Editor, Generic 359-360
- patch names 358
- PAUSE button 93-94, 224
- PBEN (Pitch Bend) event 177-178, 256
- Per: slider 260
- PERFDIR= tool type 414
- performance files 50, 52
- PERFTOOL= tool type 413
- permanent memory, synth 347
- PGM (Program Change) event 63, 145, 176, 256
- PGM: slider 176
- pgm= 256
- pitch 157, 161, 162, 169, 208
- Pitch Bend (PBEN) event 146, 177-178, 256
- pitch bend sensitivity 178
- pitch bend wheel 173
- Pitch Bend, Display 146
- Pivot Value: 308
- Pixel 431
- play buffer 347, 356
- PLAY button 24, 25, 95, 224
- Play in Context 218, 221
- Play Note, from Keymap 309
- PLAY PARAMETERS window 216, 268
- Play Sequence (PSEQ) 116, 187-190
 - from Keymap 310
- Point 431
- Pointer 431
- Poly Mode 287
- Polyphonic 431
- Polyphonic A.T., Display 145
- Polyphonic Aftertouch (PAT) events 63, 145, 157, 171-173, 254-255
 - scaling 160
- polyrhythms 215
- Pop-out Menu 431
- Portamento On/Off 286
- Prefix:, SUB-STRING Panel 371
- preroll 106, 431
- press= 254, 255
- Pressure, Channel Aftertouch 171, 254
- Pressure, Polyphonic Aftertouch 173, 255
- Pressure: slider 254
- PREVIEW button 101, 121-122, 356-357
- Preview Program: 367
- Program 431
- Program Change 63, 175-176
 - from Keymap 312
- Program Change event 145, 256
 - display 145
- program disk 3, 50
- program name, title bar 278
- Program number 176, 256, 346, 358, 431
- Program: control, Librarian 355-356
- Program: slider 256
- Prompt 431
- protecting, disk 3
- Protocol Editor 27, 363-387
- Protocol Language 372-387
- protocol name 346
- Protocol Name: 367
- protocols 372, 431
 - dialogue model 374
 - generic 354
 - messages 372
 - types 351
- PROTODIR= tool type 414
- PSEQ (Play Sequence) events 116, 187-190
- Punch 432
- Punch Overlay 103-104
- Punch-In 60, 99, 101, 104, 226
 - Auto 101, 106, 180
 - Manual 99-100, 106, 180
- Punch-In/Out, from Keymap 313

- Punch-Out 61, 104
- punched 118
- Quantize 146
- QUANTIZE button, Quantizer 152
- Quantizer Module, Bar Editor 146-152
- Quarter frame 432
- Quick list 432
 - Channel: 249, 251, 269
 - Event Type: 249
- Quiet tool word 210-211, 266
- QUIET= tool type 413
- quintuplets 215
- Quit, Music-X 49
- R. Velocity slider 253
- RAD: 36
- RAM 3, 5, 36, 113, 402, 432
 - requirements 3
- RAM: disk 5
- ramp, scaling modules 154, 158, 160, 184
- Random Access Memory (see RAM)
- Random Factor slider, Aftertouch Scaling 159
- Random Factor slider, Velocity Scaling 155
- RANGE button 301
- Range Mode, Keymap Editor 303-306
- Range: indicator 308
- ranges, keymap 294, 301, 302
- RE-OUT 288-289, 432
- real time 111
- Recall (sequence) 132-133
- receive 432
 - data formats 350
 - MIDI Start & Stop 79
 - modes, setting 285
- RECEIVE button 350-355
- RECEIVE Panel, Protocol Editor 369-370
- Record Bars 101, 106
- Record Buffer 28, 122, 128
- RECORD button 96-97, 225
- RECORD SEQUENCE Window 96-97, 98-106, 114
- Recording 314-316
- Recording Modes 225-232
- Recoverable Amiga Drive (see RAD:)
- Relative Clock Rate 182, 259
- Relative Sequence 102, 164, 179, 212, 217, 222, 223, 248
 - tempo of 259
- Relative Time 432
- release velocity 152, 153, 155, 169
 - display 145
 - scaling 155, 226
- Remap panel 281-285
- Remap T/S button 219-220
- Remove tool word 202, 264-265
- Rename
 - disk 6, 40
 - directories and files 40
- Repeat tool word 220-221, 268
- Repeats (REPT) event 114, 180-181
- Repeats parameter 258
- Repeats: slider 258
- REPT (Set Repeats) event 114, 258
- Requester 392-393, 432
- Reset Controllers button 287
- resolution 72, 432
- Reverb envelope 333
- Rewind button 94
- Rhythmic Offsets 399-400
- ROM 432
- RPT: slider 181
- RS-232 pass through 16
- RVL: slider 169
- S+ and S- 331
- safety prompt
 - Edit 119-120
 - Exit 133
 - New 130, 131
- Sample 320, 432
- Sample dump 433

- Sample List 325-327
- Sample Name 326
- Sampler 433
- Save 8, 29, 31, 37, 433
 - Filter 279
 - Keymap 297-298
 - Library 341
 - Performance 29, 52-56
 - Protocol 366-367
 - Sample 322-323
 - Sequence 57, 130
- Scale Aftertouch 158
- SCALE button, Aftertouch Scaling 160
- SCALE button, Velocity Scaling 156
- Scale Velocity 152, 153
- Screen 433
- screen colors 168
- Scroll 161, 163
- Scroll Arrows, Envelope Editor 331
- scroll bar 34
- Scroll tool word 211, 267
- Segmented 433
- select (sequence) 131
- Select All 142-143, 246
- SELECT SEQUENCE TO COPY TO window 61
- SELECT SEQUENCE TO EDIT window 131
- SELECT SEQUENCE TO MERGE TO window 63
- SELECT SEQUENCE TO STORE TO window 132
- Select tool word 127, 138, 204-205, 265
- selected events 205
- Selected Events button, Quantizer 151
- Send 433
- SEND button 346-350
- SEND panel, Protocol Editor 370-371
- Sending & Receiving, protocol strings 372-375
- Seq field (Track Window) 117
- seq field, Sequence List 113
- Seq, Sequence List 112-113
- Seq: indicator 270
- SEQ: slider 188, 191, 192
- seq= 260, 262
- sequence 127, 262, 433
 - audition 216
 - composite 202
 - current 28, 112, 122, 132
 - display 247
 - editing 161
 - editors 127
 - length of 113
 - making 66
 - memory 118
 - new 28-29
 - target 99, 103
- Sequence List 28, 112-116
- Sequence Name 66, 114
- Sequence Number (Seq) 112-113
- Sequence: name (title bar) 129
- Sequence: slider 260, 262
- sequencer 46, 433
- Sequencer Page 45-122
- Sequences (subfile switch) 53
- Sequences1-61.Kmap 315
- Serial 433
- serial port 16, 48
- Set All 283
- Set Author Name 57
- SET button 93, 301
- SET column, Channelizer 58
- Set Display 342
- Set Filters 26
- SET GRID SIZES window 213-214
- Set Punch-In 60
- Set Punch-Out 61
- Set Repeats (REPT) event 114, 180-181, 221, 258

- SetSig tool word 196-197, 263
- Sforzando envelope 333
- Shell 91
- SHOW button 300
- Show Clip 143, 246
- Show Mode, Keymap Editor 306-307
- SIG: 184
- single patch buffer 356
- single patch data, writing 348
- Size, Sample List 326
- Slider 234-235, 433
- SMPTE 21, 69, 433
 - conversion to MIDI Time Code 75
 - reader 21, 76, 433
 - unit 75
- Snap 165, 168, 171, 172, 174, 176-181, 183, 185, 187-189, 191-196, 206
 - definition 208-210
- Software 433
- Solo Sequence (SSEQ) 114, 191-193, 262
- Solo Track (STRK) 114, 195-196
 - from Keymap 312-313
- Song Assembly 403-410
- Song Position 85-86, 87
- Song Position Delay 86-90
- Song Position Pointer messages 85, 86, 217
- Song Title 47, 54
- SongStart.Perf 28
- SONIX (samples) 322
- sound module (external) 20, 433
- source sequence 61
- Special Commands, protocol language 386-387
- Splice (SPLI) event 102, 179-180, 258
- SSEQ (Solo Sequence) 114, 191-193, 262
- Start of System Exclusive byte 257
- Start Only button, Quantizer 149
- Start Sequencer, from Keymap 313
- Start Time 250
 - editing 249
 - events 179, 248, 251
- Start+Duration button, Quantizer 150
- Start+Stop button, Quantizer 150
- Start: offset slider 218-219
- STEP and BACK 135, 227, 228, 233-234, 269
- Step recording 135, 269, 225, 227-232, 403
- STOP button 24, 25, 97, 225
- Stop event 181-182, 258
- Stop Only button, Quantizer 149
- Stop Sequencer, from Keymap 313
- Stop Time
 - display 246
 - events 252, 246
- stop= 252
- storage 36
- Store (sequence) 132
- Store 434
- STORE 62, 118-119, 120, 134
- STORE TO WHERE? window 62
- string gadget 37
- String Variables, protocol language 383-386
- strings, protocol language 372
- stripe 75, 76, 434
- STRK (Solo Track) 114, 195-196
- sub-strings 365
- SUB-STRINGS Panel 371
- subfile switches 52-56
- subfiles 50, 51
- Support 434
- Suspend Music-X 48
- Sync 434
- Sync Menu 68-90
- Sync Settings 55
- synchronization 21, 68, 82, 84, 166
- Synthesizer 434
- SysEx (see also System Exclusive) 434

- System Exclusive 178, 364-365
 - codes 372
 - events 63, 64, 146, 166-167, 233, 249, 250-251, 256-257
 - display 146
 - extracting 65
 - formats 372
 - limitations of 375-376
 - messages 364
- SYSX (see System Exclusive)
- T= field 141, 142, 165, 168, 172, 174, 176, 177, 179, 180, 181, 183, 185, 187, 188, 191, 192, 194, 195, 206, 222
- Target 434
- target filter 279
- target sequence 99, 103
 - default 131
- TEMP (Tempo Change) event 259
- template 56
- tempo 182
- Tempo Change event 109, 114, 182-184, 259
- tempo changes 113
- tempo range 109
- tempo setting 56
- Tempo slider 28, 109
 - from Params 216-217
- Tempo Window 109-110
- temporary area 356
- text (boxes/gadgets) 37
- text editing cursor 114
- Thru Box 16, 434
- Thumb Index 13
- timbre 153, 157
- Time Code 21, 75, 77, 79-81
- Time Code Offset 82-83
- Time field adjustment 263
- Time field, Event List 248-249
- Time field, Sequence List 114
- time keeping 68-69
- time line 211, 221
- Time Parameters 56
- time signature 56, 111
 - global 111, 184
 - current 264
- Time Signature Change (TSIG) event 111, 114, 161, 184-185, 259-260
- Time Signature control 220
- time signature indicator 197, 264
- Time Signature window 111
- Time Stamp 434
- time stamping 96
- Time Step 228
 - from Keymap 313
- time/pitch graph 128, 161
- time
 - Absolute 69, 71, 80, 99, 106, 107, 114
 - Relative 69, 70, 107, 114
 - musical 161
- Time: clock 81
 - changing 107-108
- timer chips 70
- timing of events 146
- Title Bar 46, 129, 241, 278, 296, 296, 321, 339, 366
- TMP: slider 259
- TO column 58
- Toggle 435
- Tool List 196, 263
- Tool Words 196-221, 263-268
- Track 435
- track number 194
- Track window 117-118
- Track: slider 262
- tracks 117
- transposition 189, 260
- transfer, patch data 346
- Transmit (MIDI Start & Stop) 78

Transport Controls 223-225, 270
Transport Window 92-97
Transpose: slider 261
Transposition 189, 399
 indicator 223
 number 189
 Play Sequence 260-261
 transposition factor 141, 142
Trigger 435
triplets 215
TRK: slider 194, 195
trk= 262
trns= 260
TSIG (Time Signature Change) event 259-260
Tuning, Amiga Samples 327, 328
tuplet numbers 213, 215
Tutorials 389-397
Type field 240, 249
Type, Install Module 418
UNDO button 240, 298-299
Undo tool word 266
Unmark tool word 142, 204, 207, 245, 266
Upload 435
Use Original Program #'s 345
Use Zero Origin 67, 248
utilities 3
val= 256
Value: slider 256
variables, protocol language 379-382
VBI 74, 435
velocity 152, 154, 155, 159
Velocity Scaling Module 152-156
Vertical Blanking Interval (VBI) 74
Vibrato On/Off 286
video 21
Video Clock 74, 217
Virtual 435
virtual sliders 255, 271
Voice 435
volume 153, 157
Wild Card Character, protocol language 379
Window 435
Workbench 4-8, 39-40, 49-50, 67-68, 91
 Tool Types 410-415
wrap around 189
Zero Origin 248
zoom 143, 165, 206
Zoom Arrows, Envelope Editor 331
Zoom+ and Zoom- tool word 207-208

Bibliography

MIDI 1.0 Detailed Specification

The International MIDI Association

5316 W. 57th St.

Los Angeles, CA 90056

(213) 649-6434

(This is the official MIDI document prepared as a joint effort between the MIDI Manufacturers Association (MMA) and the Japan MIDI Standards Committee (JMSC) to explain the MIDI 1.0 specification.)

SMPTE/EBU Longitudinal & Vertical Interval Time Code

EECO Incorporated

1601 East Chestnut Avenue 92702

PO Box 659 Santa Ana California

(714) 835-6000

(A short history and explanation of SMPTE/EBU time code.)

EA IFF 85

Commodore Business Machines, Inc.

attn: C.A.T.S.

1200 Wilson Dr.

West Chester, PA 19380

(This is the documentation for the Interchange File Format. It is available from Commodore Amiga Technical Support (C.A.T.S.). The relevant section is FORM 8SVX, which describes the format of 8 bit samples.)

The MIDI System Exclusive Book

Scacciaferro, Joe

Ferro Technologies

Third Earth Publishing Inc., Pompton Lakes, N.J.

ISBN 0-88188-586-X

(A collection of MIDI implementation charts for many instruments.)

COPYRIGHT INFORMATION

The enclosed software product is copyrighted and all rights are reserved by MICROILLUSIONS. It is published exclusively by MICROILLUSIONS. The distribution and sale of this product are intended for the use of the original purchaser only and for use only on the computer system specified. Lawful users of this program are hereby licensed only to read the program from its medium into memory of a computer solely for the purpose of executing the program. Copying (except for one backup copy on those systems which provide for it) — duplicating, selling, or otherwise distributing this product is a violation of the law.

This manual and all other documentation contained herein are copyrighted and all rights reserved by MICROILLUSIONS. These documents may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from MICROILLUSIONS.

Willful violations of the Copyright Law of the United States can result in civil damages of up to \$50,000 in addition to actual damages, plus criminal penalties of up to one year imprisonment and/or \$10,000 fine.



17408 Chatsworth St., Granada Hills, CA 91344